



Onderwijsgroep Professionele Opleidingen  
Handelswetenschappen en bedrijfskunde

# Stage: Xamarin app voor RealDolmen Education

Bachelorproef aangeboden door  
**Samuel Debruyne**  
tot het behalen van de graad van  
Bachelor in Toegepaste Informatica

Stagebegeleider Odisee: **Serge Van Cleynenbreugel**  
Stagebegeleider RealDolmen: **Nico Vermeir**

---

Academiejaar 2015 – 2016

## Projectgegevens

<b>Projectcode:</b>	ST1516-SDRDEM (vervolg op BP1516-SDRDEM)
<b>Stagebedrijf:</b>	RealDolmen
<b>Klant:</b>	RealDolmen Education
<b>Projectomschrijving:</b>	Xamarin app voor RealDolmen Education
<b>Stagementor Odisee:</b>	Serge Van Cleynenbreugel
<b>Stagementor RealDolmen:</b>	Nico Vermeir
<b>Groepswerk:</b>	neen

## Woord vooraf

Dit project was niet mogelijk geweest zonder de ondersteuning die ik uit vele hoeken gekregen heb. Om die reden zou ik graag een dankwoordje richten aan enkele mensen.

Nico Vermeir had in dit project de rol van opdrachtgever en externe stagebegeleider. Elke vraag omtrent de opdracht, zowel technisch als functioneel, werd meteen beantwoord en we konden wekelijks afspreken om de voortgang van het project te bespreken. Zijn feedback, de sprint reviews en de vlotte samenwerking hebben een grote invloed gehad in het slagen van dit project, waarvoor dank.

Serge Van Cleynenbreugel is docent bij de opleiding Toegepaste Informatica aan Odisee en doceert vakken rond softwareontwikkeling voor Windows en het ontwerp van gegevensbanken. Zijn vele ervaring hieromtrent en ook bij de opbouw van deze integratietekst is zeer nuttig gebleken. Ik ben dan ook zeer dankbaar voor alle tijd, de snelle respons en de zeer waardevolle feedback die ik van hem ontvangen heb.

Daarnaast zou ik ook graag alle docenten bedanken die me de afgelopen jaren kwalitatief onderwijs, interessante uitdagingen en ongelofelijk veel kennis bijgebracht hebben.

Tot slot wens ik ook mijn ouders en mijn vriendin te bedanken voor alle moeite die ze in mijn opleiding gestoken hebben in de vorm van ondersteuning of het nalezen en feedback geven op de vele opdrachten. Bedankt!

## Inhoudsopgave

<b>Projectgegevens .....</b>	<b>2</b>
<b>Woord vooraf .....</b>	<b>3</b>
<b>1 Inleiding .....</b>	<b>7</b>
<b>2 Voorstelling van de opdrachtgever .....</b>	<b>8</b>
2.1 RealDolmen .....	8
2.2 RealDolmen Education .....	8
2.2.1 <i>Instructor-Led Training</i> .....	9
2.2.2 <i>E-Learning Projects</i> .....	9
2.2.3 <i>Technical Writing</i> .....	9
<b>3 Beschrijving van de opdracht .....</b>	<b>10</b>
<b>4 Aanpak .....</b>	<b>11</b>
4.1 Opstart van het project .....	11
4.1.1 <i>Vereiste installaties</i> .....	11
4.1.2 <i>Configuratie van broncodebeheer en continue integratie</i> .....	11
4.2 Projectmatige aanpak .....	12
4.3 Sprint 1 .....	13
4.4 Sprint 2 .....	15
4.5 Sprint 3 .....	17
4.6 Sprint 4 .....	19
4.7 Sprint 5 .....	21
4.8 Sprint 6 .....	23
<b>5 Besluit .....</b>	<b>25</b>
<b>6 Zelfreflectie .....</b>	<b>27</b>
<b>7 Figuren en tabellen .....</b>	<b>28</b>
<b>8 Bibliografie .....</b>	<b>30</b>
<b>9 Bijlagen .....</b>	<b>32</b>
9.1 Tijdsbesteding .....	32
9.1.1 <i>Werkuren</i> .....	33
9.1.2 <i>Evenementen</i> .....	35
9.1.3 <i>Opstellen documentatie</i> .....	36
9.2 Weekverslagen .....	37
9.2.1 <i>Week 1</i> .....	37
9.2.2 <i>Week 2</i> .....	38
9.2.3 <i>Week 3</i> .....	39

9.2.4	<i>Week 4</i> .....	41
9.2.5	<i>Week 5</i> .....	43
9.2.6	<i>Week 6</i> .....	47
9.2.7	<i>Week 7</i> .....	49
9.2.8	<i>Week 8</i> .....	50
9.2.9	<i>Week 9</i> .....	51
9.2.10	<i>Week 10</i> .....	52
9.2.11	<i>Week 11</i> .....	54
9.2.12	<i>Week 12</i> .....	55
9.3	Evenementverslagen .....	56
9.3.1	<i>Git and Beyond</i> .....	56
9.3.2	<i>Microsoft Build 2016</i> .....	58
9.3.3	<i>Xamarin meetup</i> .....	60
9.3.4	<i>Global Azure Bootcamp 2016</i> .....	62
9.3.5	<i>Xamarin Evolve</i> .....	64
9.4	POP / competenties .....	66
9.4.1	<i>Begin van de stage</i> .....	66
9.4.2	<i>Einde van de stage</i> .....	67
9.4.3	<i>Overzicht scores</i> .....	68
9.4.4	<i>Technische concepten</i> .....	68
9.4.5	<i>Conclusie</i> .....	69
9.5	Termenlijst.....	70
9.6	Architectuur .....	71
9.6.1	<i>MVVM Light Toolkit</i> .....	71
9.6.2	<i>Gedeelde code</i> .....	72
9.6.3	<i>Windows Phone app</i> .....	72
9.6.4	<i>Android app</i> .....	72
9.6.5	<i>iOS app</i> .....	73
9.6.6	<i>Overzicht</i> .....	73
9.7	Gebruikte design patronen.....	78
9.7.1	<i>Model-View-ViewModel (MVVM)</i> .....	78
9.7.2	<i>Dependency Inversion</i> .....	80
9.7.3	<i>Observable-Observer</i> .....	81
9.7.4	<i>Builder</i> .....	82
9.7.5	<i>Command</i> .....	82
9.7.6	<i>Overige</i> .....	82
9.8	Caching.....	83

9.8.1	<i>Frequente problemen bij mobiele applicaties</i>	83
9.8.2	<i>Werking van de cache</i>	83
9.8.3	<i>Technische werking</i>	84
9.9	Authenticatie	86
9.9.1	<i>Structuur van de code</i>	87
9.9.2	<i>Het gebruik van een Security Token Service</i>	87
9.9.3	<i>Windows Phone</i>	88
9.9.4	<i>Android en iOS</i>	89
9.10	Logging	90
9.11	Vertalingen	92
9.11.1	<i>Vertalingen bij Android</i>	92
9.11.2	<i>Vertalingen bij iOS</i>	92
9.11.3	<i>Vertalingen bij Windows Phone</i>	93
9.11.4	<i>Cross-platform vertalingen</i>	93
9.12	Testing	95
9.12.1	<i>Mocking library: Moq</i>	95
9.12.2	<i>Test coverage</i>	96
9.13	Visueel ontwerp	97
9.13.1	<i>View lifecycle management</i>	97
9.13.2	<i>Windows Phone</i>	98
9.13.3	<i>Android</i>	98
9.13.4	<i>iOS</i>	99
9.14	Ontwikkelomgeving	101
9.14.1	<i>Microsoft Visual Studio en Xamarin voor Windows</i>	101
9.14.2	<i>Xamarin Studio voor Mac</i>	101
9.14.3	<i>Broncodebeheer (VCS): TFS en Git</i>	102
9.14.4	<i>Continue integratie met Visual Studio Team Services (VSTS)</i>	102
9.15	Afhankelijkheden	104
9.15.1	<i>Xamarin Components</i>	104
9.15.2	<i>NuGet packages</i>	105
9.16	Broncode	111
9.17	Presentaties	112
9.17.1	<i>MS Community</i>	112
9.17.2	<i>Slotpresentatie RealDolmen</i>	113
9.17.3	<i>Slotpresentatie Odisee</i>	116

## 1 Inleiding

Dit tweede deel van de bachelorproef bestaat uit een uitgebreid verslag van de stage en het gerealiseerde project. Het eindproduct van het project is een mobiele applicatie voor Android, iOS en Windows Phone aan de hand van Xamarin. Het doel was om bij de ontwikkeling van de applicatie zoveel mogelijk code te delen over deze drie mobiele platformen.

Naast de vereisten van de klant werd er ook gestreefd naar een kwalitatief resultaat door middel van nieuwe technische concepten en een solide architectuur. Deze hebben als doel duidelijkheid te scheppen in de gecreëerde broncode en volgen herkenbare patronen die andere softwareontwikkelaars zonder moeite kunnen interpreteren.

Daarnaast is ook het organisatorische luik belangrijk. Zonder projectmatige aanpak op basis van erkende werkmethodes zoals scrum, loopt elk project verloren. De theorie hieromtrent die tijdens de opleiding Toegepaste Informatica werd aangeleerd, kon tijdens de stage op de proef gesteld worden.

Tijdens de stage vonden relevante evenementen plaats met technische sessies. De verslagen van deze evenementen en hun toepassingen zijn toegevoegd als bijlage 9.3 EVENEMENTVERSLAGEN.

Tot slot was het de eerste keer dat een project van dergelijke omvang uitgevoerd werd en dit in een professionele context met meerdere stakeholders. Dus ook op communicatief vlak en professionalisme was deze stage een eerste, maar uitgebreide kennismaking.

In dit document worden verschillende termen gebruikt die niet voor alle lezers even duidelijk zijn. Deze termen werden verduidelijkt in bijlage 9.5 TERMENLIJST.

## 2 Voorstelling van de opdrachtgever

### 2.1 RealDolmen

RealDolmen is een Belgisch bedrijf gespecialiseerd in de integratie van bestaande en eigen ICT-producten. Het bedrijf is ontstaan in 2008 na de fusie van Real Software en Dolmen en stelt vandaag meer dan 1500 mensen te werk. Het heeft afdelingen die elk gespecialiseerd zijn in een specifiek technisch domein, zoals bijvoorbeeld .NET software, Java software, netwerken en systemen, business intelligence...



Figuur 1: logo van RealDolmen

De hoofdzetel van het bedrijf is gevestigd in Huizingen. Daarnaast heeft het satellietkantoren in Kontich, Diegem, Harelbeke, Lummen, Namen en Bergen. Ook in Luxemburg is er nog een kantoor van Real Solutions.

Dankzij het goed gespreide kantorennetwerk is RealDolmen heel actief in lokale markten. Hier tracht het de referentie te zijn voor geïntegreerde en volledige ICT-oplossingen. Bij RealDolmen staat het eindproduct centraal, niet de technologie erachter. Zo kunnen de klanten zich concentreren op hun eigen kerntaken en wordt ICT een middel om die efficiënter te laten verlopen.

In dit project heeft RealDolmen de rol van opdrachtgever. De stage wordt door de .NET-afdeling van het bedrijf begeleid, maar de eigenlijke klant is een ander intern departement: RealDolmen Education.

### 2.2 RealDolmen Education

Education is een departement van RealDolmen dat opleidingen verzorgt voor zowel eigen personeel als medewerkers van externe bedrijven.

*Wat is goede software waard indien uw eindgebruikers er niet of niet efficiënt mee kunnen werken? Hoe goed zal uw eigen infrastructuur voldoen aan de eisen van uw business indien uw systeembeheerders niet mee evolueren met de laatste trends?  
Hoe goed zal uw eigen software voldoen aan de eisen van uw business indien uw ontwikkelaars niet mee evolueren met de laatste trends?*

*RealDolmen Education*

Het departement biedt verschillende diensten aan, maar de drie kerntaken zijn Instructor-Led Training (ILT), e-Learning Projects en Technical Writing. Vanzelfsprekend passen alle diensten en vormen van opleidingen binnen het technische kader waarin RealDolmen actief is op de markt.

### 2.2.1 Instructor-Led Training

De klassieke opleidingen met een lesgever en een groep studenten vormen de belangrijkste troef van RealDolmen Education.

### 2.2.2 E-Learning Projects

Naast ILT ontwikkelt RealDolmen ook digitale leeromgevingen waarin studenten zelfstandig materie kunnen aanleren, begeleid door software.

### 2.2.3 Technical Writing

Vaak ontbreekt het software aan goede documentatie. Hier probeert RealDolmen Education een oplossing voor te bieden. Ze kunnen zowel technische handleidingen als educatief materiaal om met software aan de slag te gaan opstellen. Dit kan zowel voor bestaande als voor software uit eigen huis.

### 3 Beschrijving van de opdracht

RealDolmen bundelt jaarlijks een tiental interessante stageopdrachten waarbij de klant telkens een departement van RealDolmen is, maar niet noodzakelijk hetzelfde departement als hetwelk dat de opdracht begeleidt. Zo vraagt het Education departement naar een mobiele applicatie voor meerdere platformen.

RealDolmen Education biedt ILT-opleidingen aan via de website [education.realdolmen.com](http://education.realdolmen.com). Deze website laat bezoekers toe om te zoeken in het aanbod aan opleidingen, zowel via een zoekfunctie als door gebruikers toe te laten opleidingen gecategoriseerd weer te geven. De website bevat ook nog andere functies zoals een contactformulier en een mogelijkheid om feedback op een opleidings-sessie te geven.

The screenshot shows the homepage of the RealDolmen Education website. At the top, there is a logo with a blue square containing a white stylized 'R' and the text 'REALDOLMEN' in blue and red. Below the logo is the tagline 'We make ICT work for your business'. A language switcher shows 'EN NL FR'. A navigation bar at the top has links for 'Home', 'Kalender', 'Evaluatie', 'Contact', and 'Over ons'. On the left side, there is a sidebar with a 'Zoek' (Search) input field and a red 'Zoek' button. Below the search is a section titled 'Opleidingen' (Education) with a list of categories: 'Information Workers' (Microsoft Excel, Word, PowerPoint, Access, Outlook, SharePoint, Visio, Project, Windows, Social Media, Adobe, Other), 'Open Brains Sessions' (DBA, Web Development), 'Microsoft Azure' (IT Pro, Developer, CxO), and 'Methodologies' (ITIL, BPMN, Project Management, Lean Six Sigma, Agile and Scrum). The main content area features a red banner with the text 'Welkom bij RealDolmen Education.' Below the banner, there is descriptive text about the website's purpose and a contact form. To the right, there is a section titled 'Nieuwe cursussen' (New courses) with four course cards: 'Agile for business analysts' (no description), 'Agile pm foundation' (no description), 'Agile pm foundation and practitioner' (no description), and 'APMG Lean IT' (no description).

Figuur 2: de website van RealDolmen Education

Momenteel is de website niet geschikt voor gebruik op mobiele toestellen zoals smartphones en tablets. Navigeren door de categorieën verloopt moeizaam en de gebruikerservaring van de website is op deze toestellen niet vergelijkbaar met die op een desktopcomputer of laptop.

Het departement wil daarom een mobiele toepassing die soelaas brengt aan deze tekortkomingen. Het zorgt ook meteen voor een duwtje in de rug voor de naamsbekendheid van RealDolmen in applicatiewinkels op mobiele platformen. Om het potentiële aantal gebruikers van de toepassing te maximaliseren, wil RealDolmen Education de applicatie tegelijk op de populairste mobiele platformen lanceren.

De vraag van de klant werd verder toegelicht in het eerste deel van de bachelorproef.

## 4 Aanpak

### 4.1 Opstart van het project

#### 4.1.1 Vereiste installaties

De stage begon met het installeren van het vereiste werk materiaal. Dit ging om de volgende zaken:

- Laptop met Microsoft Windows 8 of nieuwer, Microsoft Visual Studio 2013 of nieuwer, laatste versie van Xamarin en voldoende werkgeheugen voor het testen van de te ontwikkelen applicaties
- MacBook met de laatste versie van Xamarin
- Visual Studio Team Services
- Source code repository

Door Wi-Fi-problemen met de Windows-laptop moest de installatie van alle software twee keer gebeuren wat al gauw een volledige dag in beslag nam. De laptop was oorspronkelijk eveneens niet voorzien van voldoende werkgeheugen. Dit valt ook na te lezen in 9.2.1 WEEK 1 en 9.2.2 WEEK 2.

#### 4.1.2 Configuratie van broncodebeheer en continue integratie

De volgende stap was het instellen van broncodebeheer. Hier opteerde RealDolmen voor Visual Studio Team Services. Initieel werd een systeem opgezet met Microsoft Team Foundation Services, maar door ontbrekende compatibiliteit met Apple OS X, werd nadien een broncodesysteem met Git opgezet.

Vervolgens werd een systeem voor continue integratie opgezet. Dit zorgt ervoor dat een ontwikkelaar meteen geautomatiseerde feedback kan krijgen op code die hij net geschreven heeft. Er kwam wat extra werk kijken bij het instellen van de iOS app.

De gedetailleerde configuratie en verdere verduidelijking bij dit proces valt na te lezen in bijlage 9.14 ONTWIKKELOMGEVING.

## 4.2 Projectmatige aanpak

Bij de stage was het niet alleen de bedoeling om technische kennis en ervaring op te doen rond softwareontwikkeling met Xamarin, maar ook om de theorie rond projectmatige aanpak uit de opleiding in de praktijk om te zetten. Zodoende kon het project op een georganiseerde manier aangepakt worden waarbij alle partijen steeds op de hoogte waren van de voortgang van het project.

Er werd gewerkt in zes sprints van elk twee weken. Heel af en toe werden de activiteiten opgeschort wegens ziekte, activiteiten op de hogeschool of bij RealDolmen... 's Avonds en op een zaterdag waren er ook evenementen met technische sessies rond onderwerpen die tijdens de stage aan bod kwamen. De verslagen van deze evenementen staan onder [bijlage 9.3 EVENEMENTVERSLAGEN](#).

Visual Studio Online (wordt ook vaak Visual Studio Team Services of VSO/VSTS genoemd), is de dienst die gebruikt werd voor het opstellen van de sprint planningen. Aan de start van het project wordt een backlog opgesteld met alle functies die de klant gevraagd heeft en ontwikkeld moeten worden. Elk onderdeel van deze lijst krijgt een prioriteit en inschatting van het werk toegewezen.

Aan het begin van elke sprint worden enkele items uit de lijst gekozen die tijdens die sprint afgewerkt moeten worden. Deze worden elk onderverdeeld in concrete, gedetailleerde taken die nooit langer dan een dag of zelfs enkele uren tijd in beslag nemen. Er wordt bij elke taak opnieuw een inschatting gemaakt van het aantal uren werk.

Tijdens de sprint zelf verhuist elke taak van de kolom 'to-do', naar 'in progress', om vervolgens onder 'done' te belanden. Zo kan VSO steeds automatisch berekenen hoe efficiënt het team werkt, of de inschattingen stroken met de werkelijkheid en een *burndown chart* opstellen. Deze grafiek laat zien hoe snel het werk vooruit gaat en hoe snel het werk idealiter zou moeten vooruitgaan.

Ten slotte wordt al deze informatie aan het einde van de sprint geëvalueerd om inschattingsfouten, ontwikkelproblemen en tekorten of overschotten aan tijd bij de volgende sprints te vermijden. Tijdens de stage bleek dit ook effectief te werken want de inschattingsfouten namen tijdens het verloop van de stage telkens wat meer af. De uiteindelijke tijdsbesteding is terug te vinden onder [9.1 TIJDSBESTEDING](#).

## 4.3 Sprint 1

**Data:** 1 februari – 12 februari

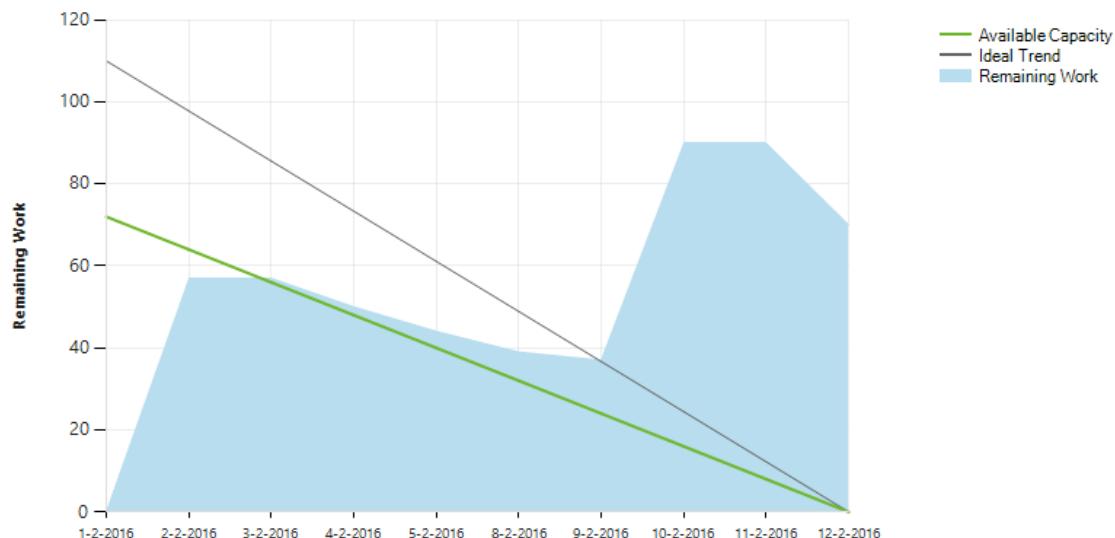
**Weekverslagen:** 9.2.1 Week 1 en 9.2.2 WEEK 2

**Evenementen:** geen

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Ik wil als geïnteresseerde een splash screen zien	
Task	Installeren benodigde tools en opbouw projecten	
Task	Tools installeren en verbinding met MacBook	
Task	Splash screen Android aanmaken	
Task	Splash screen Windows Phone aanmaken	
Task	Hoofdscherm Android aanmaken	
Task	Hoofdscherm Windows Phone aanmaken	
Task	Base classes voor http client aanmaken	
Task	Huisstijl op Android aanmaken en integreren	
Task	Huisstijl op Windows Phone aanmaken en integreren	
Task	CI-tools opzetten	Deployment
Product Backlog Item	Ik wil als geïnteresseerde een lijst van alle categorieën kunnen ophalen	
Task	Code om categorieën uit API op te halen	
Task	Lijst voor categorieën aanmaken op Android	
Task	Lijst voor categorieën aanmaken op Windows Phone	
Task	Categorieën weergeven in lijst op Android	
Task	Categorieën weergeven in lijst op Windows Phone	Development
Task	Caching voorzien om categorieën te bewaren	
Product Backlog Item	Ik wil als geïnteresseerde me kunnen inloggen en zo kenbaar maken als gebruiker	
Task	Scherm maken om in te loggen op Android	
Task	Scherm maken om in te loggen op Windows Phone	
Task	Client voor IdentityServer installeren en configureren	

Tabel 1: planning sprint 1



Figuur 3: burndown chart sprint 1

Tijdens de eerste sprint viel er een dag weg wegens ziekte en ging er ook wat tijd verloren aan configuratieproblemen met de laptop. Zoals zichtbaar op de burndown chart in Figuur 3 werden er unit tests toegevoegd aan het project waardoor de hoeveelheid werk meteen toenam. Ook omdat de vereiste MacBook pas later beschikbaar kwam, werd er wat geschoven met de planning en werd het werk aan de iOS app uitgesteld.

De uitdaging bij de eerste sprint was om een goede architectuur voor het project op te zetten. Het werd vrij snel duidelijk dat een MVVM-structuur (zie 9.7.1 MODEL-VIEW-VIEWMODEL (MVVM)) zich hier het best toe leende. Om daarnaast de hoeveelheid code die over de drie mobiele platformen gedeeld wordt te optimaliseren, werd Autofac (9.15.2.2 AUTOFAC), een hulpmiddel voor dependency inversion (9.7.2 DEPENDENCY INVERSION), opgezet. De resulterende architectuur is gedocumenteerd in bijlage 9.6 ARCHITECTUUR.

Voor het authenticatie-aspect van de applicatie, was integratie met een STS of Security Token Service van RealDolmen gevraagd. Aangezien deze dienst nog niet beschikbaar was tijdens de eerste sprint, werd deze taak uitgesteld naar de tweede sprint. Caching kon daarentegen wel al geïmplementeerd worden. De volledige uitwerking van het cachingmechanisme is gedocumenteerd in bijlage 9.8 CACHING. Er bestaan veel uiteenlopende manieren om dit te implementeren, maar een blogpost (Gibbens, 2015) met enkele concrete en volledige voorbeelden gaf hier de doorslag.

Het splash-screen dat tijdens sprint 1 ontwikkeld moest worden, was niet zozeer een belangrijke functie voor de gebruiker, maar wel een manier om een app op te zetten. Een splash-screen is een scherm waarop enkel het logo van de applicatie weergegeven wordt. Tijdens de weergave van een splash-screen worden gegevens op de achtergrond ingeladen en kunnen de systemen zoals caching en authenticatie voorbereid worden.

Tijdens sprint 1 werd dus vooral voorbereidend werk verricht dat het verloop van de volgende vijf sprints stroomlijnde. De opgedoken problemen zorgden voor een lichte achterstand op de planning, maar konden vrij snel van de baan geruimd worden.

## 4.4 Sprint 2

**Data:** 15 februari – 26 februari

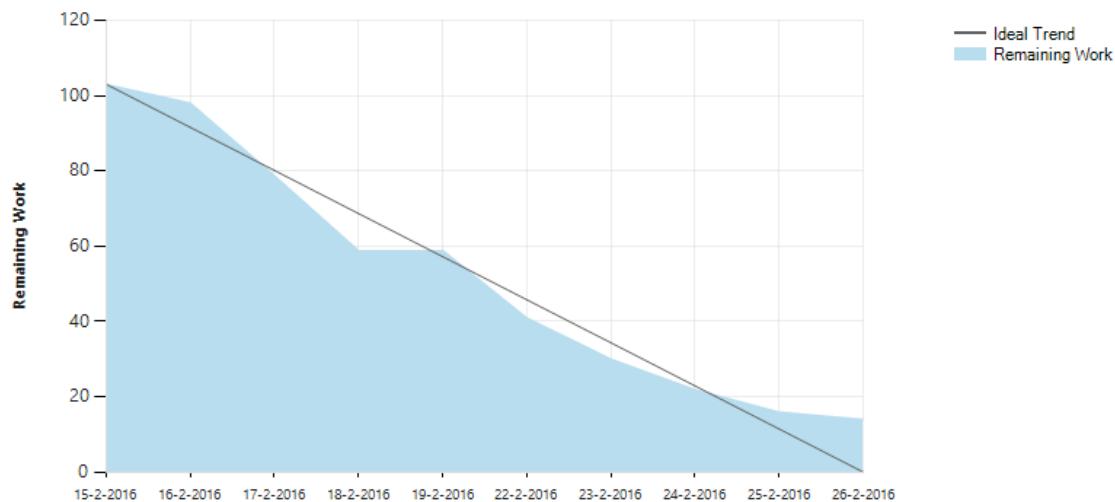
**Weekverslagen:** 9.2.3 WEEK 3 en 9.2.4 WEEK 4

**Evenementen:** 9.3.1 GIT AND BEYOND

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Ik wil als geïnteresseerde een splash screen zien	
Task	Splash screen iOS aanmaken	
Task	Hoofdscherm iOS aanmaken	
Task	Huisstijl op iOS aanmaken en integreren	
Task	iOS Development intro	
Product Backlog Item	Ik wil als geïnteresseerde een lijst van alle categorieën kunnen ophalen	
Task	Lijst voor categorieën aanmaken op iOS	
Task	Categorieën weergeven in lijst op iOS	
Product Backlog Item	Ik wil als geïnteresseerde me kunnen inloggen en zo kenbaar maken als gebruiker	
Task	Scherm maken om in te loggen op iOS	
Product Backlog Item	Onderzoek nodig voor ontwikkeling	
Task	Onderzoek naar x-platform i18n	Documentation
Task	Tests schrijven	
Product Backlog Item	Ik wil als geïnteresseerde kunnen zoeken naar een opleiding	
Task	Zoekbalk- en functie op Android	
Task	Zoekbalk- en functie op Windows Phone	
Task	Zoekbalk- en functie op iOS	
Task	API-calls voor zoeken naar opleidingen	
Task	Resultaten weergeven op Android	
Task	Resultaten weergeven op iOS	
Task	Resultaten weergeven op Windows Phone	
Product Backlog Item	Ik wil als geïnteresseerde een lijst van alle subcategorieën in een categorie kunnen ophalen	
Task	API-calls voor subcategorieën	
Task	Subcategorieën weergeven op Android	
Task	Subcategorieën weergeven op iOS	
Task	Subcategorieën weergeven op Windows Phone	

Tabel 2: planning sprint 2



Figuur 4: burndown chart sprint 2

De tweede sprint begon met een sprint review van sprint 1. Het publiek was wat groter dan bij de daaropvolgende sprint reviews en in het algemeen was de feedback positief. De aanwezigen gaven nog wat tips mee om betere sprintplanningen op te stellen die nadien ook toegepast werden en betere resultaten opleverden.

Bij de aanvang van sprint 2 was er wel een Mac beschikbaar waardoor de achterstallige ontwikkeling van de iOS app prioriteit kreeg. Ook het uitgestelde werk aan de authenticatie kon in deze sprint aangevat worden. De werking van de authenticatie is verduidelijkt in bijlage 9.9 AUTHENTICATIE.

De weergave van de hoofdcategorieën werd afgewerkt en er werd een basis opgebouwd om subcategorieën weer te geven en te zoeken naar opleidingen. Tijdens de ontwikkeling doken er enkele problemen op waarvan de oorzaak niet meteen opgespoord kon worden. Om hier een mouw aan te passen, werd logging toegevoegd aan de applicatie. Hoe dit precies in zijn werk gaat, is te lezen in bijlage 9.10 LOGGING.

Om de kwaliteit van de code te bewaken werd er verder tijd geïnvesteerd in het schrijven van unit tests voor de gedeelde code. Zo wordt de kans op bugs tijdens het uitvoeren van de applicatie kleiner en geldt dit meteen voor alle mobiele platformen. Om deze unit tests consistent te blijven uitvoeren werd een systeem voor continue integratie opgezet. Dit concept is een volledige oplossing waarmee sneller en automatisch problemen en bugs opgespoord kunnen worden, steeds stabiele versies van het product gegarandeerd zijn en om het volledige team een overzicht te geven van de stabiliteit en kwaliteit van nieuwe code. Testen werd gedocumenteerd in bijlage 9.12 TESTING en continue integratie werd uitgediept in 9.14.4 CONTINUE INTEGRATIE MET VISUAL STUDIO TEAM SERVICES (VSTS).

In het algemeen verliep de sprint ongehinderd en de opgedoken bugs werden besproken met de stagebegeleider. Deze werden aan de backlog toegevoegd en ingepland bij volgende sprints.

## 4.5 Sprint 3

**Data:** 29 februari – 11 maart

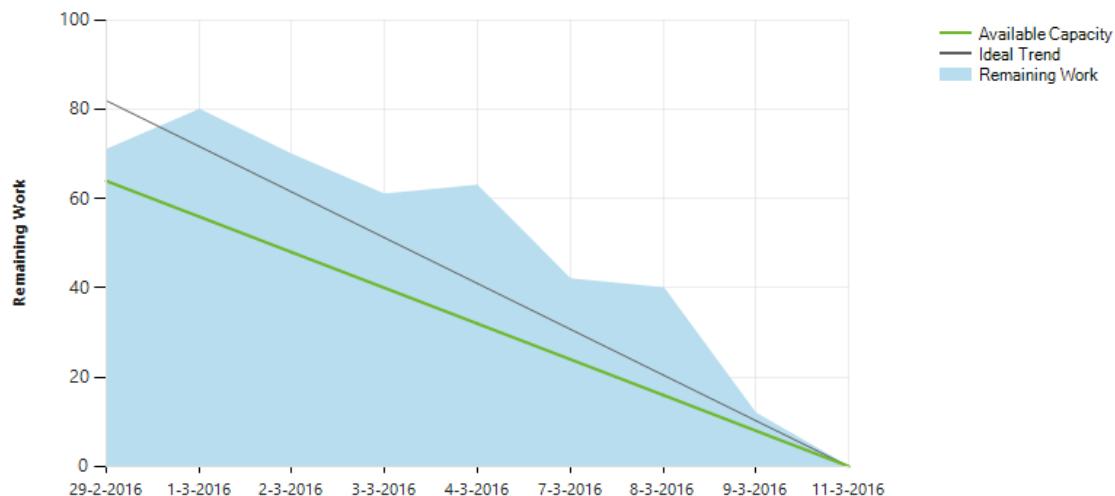
**Weekverslagen:** 9.2.5 WEEK 5 en 9.2.6 WEEK 6

**Evenementen:** jobbeurs Odisee

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Ik wil als geïnteresseerde me kunnen inloggen en zo kenbaar maken als gebruiker	
Task	API-calls voor authenticatie voorzien	Development
Task	Methodes aanmaken om authenticated API-calls te doen	Development
Task	Gegevens over ingelogde gebruiker ophalen	Development
Task	Scherm voor ingelogde gebruiker maken op Android	Design
Task	Scherm voor ingelogde gebruiker maken op iOS	Design
Task	Scherm voor ingelogde gebruiker maken op Windows Phone	Design
Product Backlog Item	Ik wil als geïnteresseerde gedetailleerde informatie over een opleiding kunnen bekijken	
Task	API-methodes voor details van een opleiding	Development
Product Backlog Item	Ik wil als gebruiker contact kunnen opnemen met RealDolmen Education	
Task	API-methodes voor contact	Development
Task	ViewModel voor contactpagina	Development
Product Backlog Item	Ik wil als gebruiker feedback over een les kunnen geven	
Task	API-methodes voor feedback	
Task	ViewModel voor feedback	
Product Backlog Item	Ik wil als geïnteresseerde alle opleidingen van een subcategorie kunnen ophalen	
Task	API-methodes voor opleidingen van categorie	Development
Task	Scherm voor opleidingen in categorie op Windows Phone	Design
Task	Scherm voor opleidingen in categorie op iOS	Design
Task	Scherm voor opleidingen in categorie op Android	Design
Task	ViewModel voor subcategorie	Development
Product Backlog Item	Ik wil als gebruiker mij kunnen inschrijven voor een opleiding	
Task	ViewModel voor inschrijving	
Bug	Bugs oplossen	Development
Task	SearchValue binding op iOS	Debugging
Task	Back-knop (navigationservice) op iOS	Debugging
Task	Loading indicator op Android	Debugging
Product Backlog Item	Ik wil als gebruiker mezelf kunnen uitloggen	
Task	Uitlogknop en bindings op Android	Design
Task	Uitlogknop en bindings op Windows Phone	Design
Task	Uitlogknop en bindings op iOS	Design

Tabel 3: planning sprint 3



Figuur 5: burndown chart sprint 3

Bij de tweede sprint review werd een kleine vertraging vastgesteld die te wijten was aan enkele bugs uit sprint 2. De oorzaak van de bugs lag hem in het gebruik van MVVM Light (zie 9.15.2.15 MVVM LIGHT TOOLKIT). Dit framework is nog niet matuur en ook documentatie over het gebruik ervan bij Xamarin apps bestaat nauwelijks. Het oplossen van de bugs werd op de backlog geplaatst.

Authenticatie sleepte nog steeds aan omdat het een complexe component is waarvan ook de backend nog niet volledig op punt stond. Tijdens sprint 3 werden de gerelateerde achterstallige taken aangepakt zodat hier verder geen tijd meer in gestoken moest worden.

Aan het begin van sprint 3 werd overgestapt naar een andere manier van ontwikkelen. De iOS-app kon namelijk enkel met behulp van een Mac afgewerkt worden en de verbinding tussen de Mac en Visual Studio was allesbehalve stabiel. Daarom werd de overstap gemaakt naar Xamarin Studio. De Windows Phone app, de unit tests en alle gedeelde code werden in Visual Studio op Windows ontwikkeld. De iOS app werd daarentegen in Xamarin Studio op Mac afgewerkt en de Android app in Xamarin Studio voor Windows. De achterliggende redenen en technische details zijn terug te vinden in bijlage 9.14 ONTWIKKELOMGEVING.

Tot slot kwam er nog een bijkomende uitdaging kijken bij de afwerking van de geselecteerde taken van sprint 3. Het werd namelijk tijd om wat meer aandacht te besteden aan *view lifecycle management*. Tijdens het gebruik van een applicatie kan elk scherm door meerdere fases gaan. Zo wordt een scherm opgeroepen, opgebouwd, weergegeven en uiteindelijk weer afgesloten en uit het geheugen gehaald. In tussentijd kan ditzelfde scherm meerdere keren verborgen en opnieuw weergegeven worden. Dit verloopt heel anders op elk mobiel platform en abstractie is niet of nauwelijks mogelijk. Hoe dit exact aangepakt werd, staat in bijlage 9.13 VISUEEL ONTWERP aangezien dit sterk samenhangt met de visuele opbouw van een scherm.

Sprint 3 verliep in het algemeen vrij vlot en ook hier werd op korte tijd veel nieuwe kennis opgedaan. De sprint moest tussentijds even hingedeeld worden omdat de mogelijkheid om RealDolmen Education te contacteren niet afgewerkt kon worden vooraleer de details van een opleiding zichtbaar waren.

## 4.6 Sprint 4

**Data:** 14 maart – 25 maart

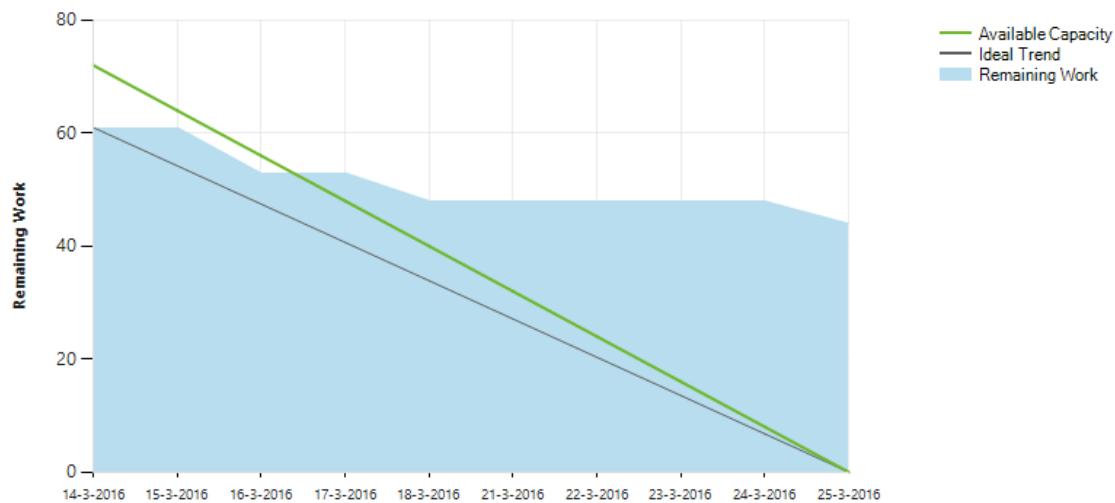
**Weekverslagen:** 9.2.7 WEEK 7 en 9.2.8 WEEK 8

**Evenementen:** geen

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Ik wil als geïnteresseerde gedetailleerde informatie over een opleiding kunnen bekijken	
Task	Scherm voor opleiding op Android	Design
Task	Scherm voor opleiding op Windows Phone	Design
Task	ViewModel voor opleiding	Development
Task	Unit tests voor ViewModel	
Product Backlog Item	Ik wil als gebruiker mij kunnen inschrijven voor een opleiding	
Task	API-methodes voor inschrijvingen	
Bug	Bugs oplossen	Development
Task	Bindings op Android updaten niet bij search	Development

Tabel 4: planning sprint 4



Figuur 6: burndown chart sprint 4

Ondanks de tussentijdse aanpassingen bij sprint 3, verliep de sprint review vlot. Bij de vierde sprint ziet de burndown chart er minder goed uit en dit valt te wijten aan een onvolledige planning. Een van de taken was het weergeven van details van een opleiding. Deze details gaan over de onderste delen van het gegevensmodel (zie Figuur 34). Er bestond vooraf maar een beperkte hoeveelheid informatie over deze gegevens en ook op de reeds bestaande website zijn deze niet meteen zichtbaar.

Er moesten enkele bijkomende schermen ontwikkeld worden om de verschillende sessies van een opleiding weer te geven die elk uit meerdere lesmomenten bestaan. De inhoud van deze gegevens is niet gedocumenteerd en de feedback van RealDolmen Education bij de slotpresentatie was dan ook dat deze gegevens niet correct geïnterpreteerd waren. Daarom is het dan ook zeer spijtig dat ze, ondanks tweewekelijkse uitnodigingen, besloten niet aanwezig te zijn bij tussentijdse sprint reviews en deze feedback niet eerder konden geven.

Bij deze sprint kwamen er opnieuw heel wat ontwerppatronen kijken bij de ontwikkeling. Deze hebben bijgedragen aan de creatie van degelijkere code en het grotendeels oplossen van de bugs uit de tweede sprint. De voornaamste en meest cruciale patronen voor de Education app zijn gedocumenteerd in bijlage 9.7 GEBRUIKTE DESIGN PATRONEN.

Een andere obstructie bij de vierde sprint was een beperking op de API die gebruikt werd om gegevens op te halen. Zoals te lezen in het verslag van 9.2.7 WEEK 7, is de onderste laag van het gegevensmodel nog niet geïmplementeerd in de API. Hiervoor moest wat code geschreven worden die testgegevens creëert zolang de API geen gegevens kan aanbieden.

In het algemeen was het werk aan de weergave van de opleidingsdetails onderschat en de opgedoken hindernissen hebben hier niet bij geholpen. Sprint 4 was de meest stroeve sprint uit het volledige project en bracht een achterstand die de planningen van sprint 5 en sprint 6 beïnvloedde. Ten gevolge was het niet langer mogelijk om eventueel bijkomende functies te implementeren die oorspronkelijk buiten de scope van het project lagen, maar wel mooi meegenomen hadden geweest.

## 4.7 Sprint 5

**Data:** 29 maart – 11 april

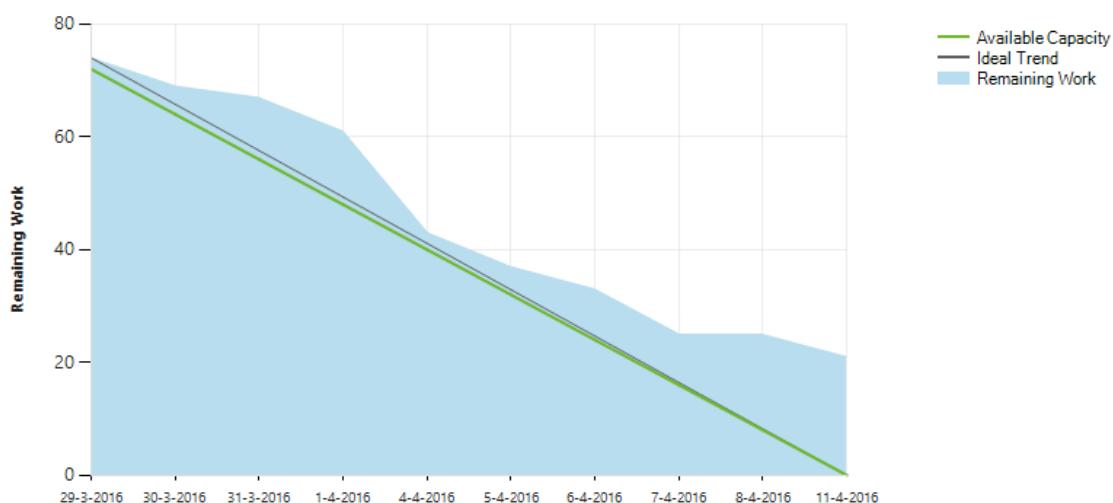
**Weekverslagen:** 9.2.9 WEEK 9 en 9.2.10 WEEK 10

**Evenementen:** MS community event RealDolmen, 9.3.2 MICROSOFT BUILD 2016 en 9.3.3 XAMARIN MEETUP

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Ik wil als geïnteresseerde gedetailleerde informatie over een opleiding kunnen bekijken	
Task	Scherm voor opleiding op iOS	Design
Product Backlog Item	Ik wil als gebruiker contact kunnen opnemen met RealDolmen Education	
Task	Contactformulier op Windows Phone	Design
Task	Contactformulier op iOS	Design
Product Backlog Item	Ik wil als gebruiker feedback over een les kunnen geven	
Task	Feedbackscherm op Windows Phone	Design
Product Backlog Item	Ik wil als geïnteresseerde alle lessen van een opleiding kunnen bekijken	
Task	API-methodes om sessies op te halen	Development
Task	ViewModel voor opleiding bijwerken	Development
Task	Lijst van sessies op Android	Design
Task	Lijst van sessies op iOS	Design
Task	Lijst van sessies op Windows Phone	Design
Task	Unit tests voor opleiding vm bijwerken	
Product Backlog Item	Ik wil als gebruiker mij kunnen inschrijven voor een opleiding	
Task	Inschrijfscherm op Windows Phone	

Tabel 5: planning sprint 5



Figuur 7: burndown chart sprint 5

De opgebouwde achterstand uit sprint 4 beheerde het grootste deel van de planning van de vijfde sprint. Dit werd ook meteen duidelijk tijdens de sprint review.

Praktisch alle taken bij de voorlaatste sprint hielden design in. De ondersteuning van Xamarin is op dit vlak nog vrij beperkt. Aan het begin van de stage waren er in Microsoft Visual Studio nog geen stabiele suggesties bij het typen van code in AXML-bestanden<sup>1</sup>. Deze suggesties verhogen de efficiëntie bij de ontwikkeling en vormen een belangrijke meerwaarde in elke programmeertaal. Na enkele updates aan het einde van de stage was dit euvel van de baan. Meer informatie over deze ontwikkelervaringen zijn terug te vinden in bijlage 9.14 ONTWIKKELOMGEVING.

Zoals beschreven in de niet-functionele vereisten in het Project Initiatie Document uit het eerste deel van deze bachelorproef, had RealDolmen gevraagd om voor elk platform een degelijke visuele weergave te ontwerpen. De nadruk lag hierbij op een ontwerp dat eigen was aan elk mobiel platform. Om dit te verwezenlijken werd bij de Android app gebruik gemaakt van Googles *Material design*. Bij de applicaties voor iOS en Windows Phone werden respectievelijk de richtlijnen van Apple en Microsoft gevolgd. Voorbeelden hiervan zijn terug te vinden in de bijhorende presentatie en in bijlage 9.13 VISUEEL ONTWERP.

Tijdens sprint 5 vond net zoals bij de meeste andere sprints een boeiend evenement plaats gerelateerd aan de stageopdracht. Alle verslagen hiervan zijn bijgevoegd als 9.3 EVENEMENTVERSLAGEN. Dit evenement in het bijzonder ging volledig over Xamarin en bevatte een sessie van Mike James, evangelist bij Xamarin Inc. Zijn inzichten hebben de aanpak van sprint 5 licht bijgestuurd om mogelijke bugs te vermijden. James is een grote voorstander van het MVVM-patroon (9.7.1 MODEL-VIEW-VIEW-MODEL (MVVM)), maar gebruikt zelf weinig tot geen frameworks of andere afhankelijkheden om dit te verwezenlijken.

Kortom dreigde sprint 5 een nogal saaie sprint te worden met veel repetitief en weinig interessant werk. Bij het visuele ontwerp kwam namelijk nauwelijks nieuwe technische ervaring kijken. Toch eindigde de sprint op een zeer positieve noot dankzij het Xamarin evenement.

---

<sup>1</sup> AXML-bestanden bevatten code die de layout en visuele weergave van Android apps beschrijven.

## 4.8 Sprint 6

**Data:** 12 april – 25 april

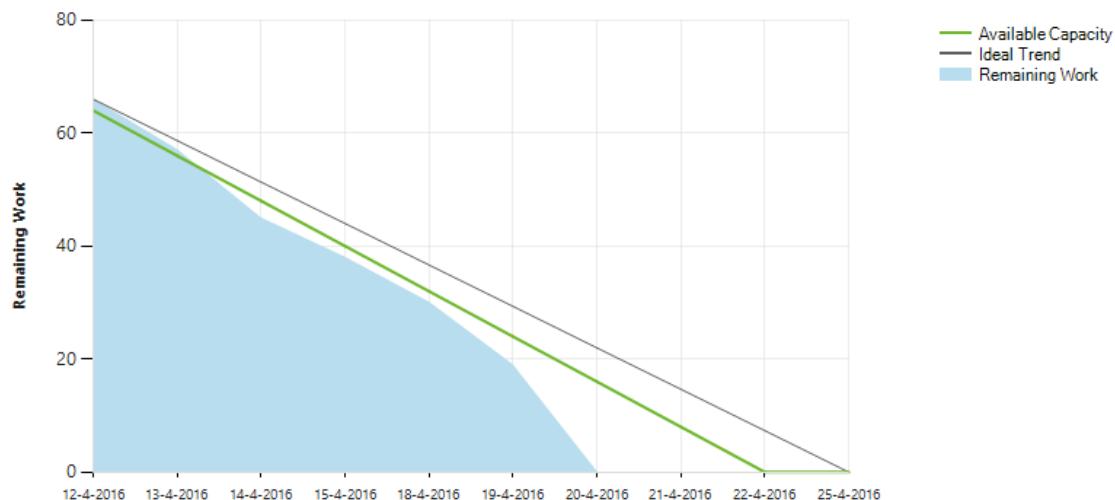
**Weekverslagen:** 9.2.11 WEEK 11 en 9.2.12 WEEK 12

**Evenementen:** 9.3.4 GLOBAL AZURE BOOTCAMP 2016 en 9.3.5 XAMARIN EVOLVE

**Sprint planning:**

Work Item Type	Title	Activity
Product Backlog Item	Slotpresentatie	
Task	Inhoud opstellen	
Task	Slides aanmaken	
	Ik wil als gebruiker contact kunnen opnemen met RealDolmen	
Product Backlog Item	Education	
Task	Contactformulier op Android	Design
Product Backlog Item	Ik wil als gebruiker feedback over een les kunnen geven	
Task	Feedbackscherm op iOS	Design
Task	Feedbackscherm op Android	Design
Product Backlog Item	Ik wil als gebruiker mij kunnen inschrijven voor een opleiding	
Task	Inschrijfscherm op Android	
Task	Inschrijfscherm op iOS	
Bug	Bugs oplossen	Development
Task	Binding issues op Android & iOS	
	Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar Nederlands	
Product Backlog Item		
Task	Vertalingen op Android	
Task	Vertalingen op iOS	
Task	Vertalingen op Windows Phone	
	Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar Engels	
Product Backlog Item		
Task	Scherm om taal aan te passen op iOS	
Task	Scherm om taal aan te passen op Android	
Task	Scherm om taal aan te passen op Windows Phone	
	Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar Frans	
Product Backlog Item		
Task	Vertalingen op Android	
Task	Vertalingen op iOS	
Task	Vertalingen op Windows Phone	

Tabel 6: planning sprint 6



Figuur 8: burndown chart sprint 6

De resterende taken bij sprint 6 hielden nog wat resterend opmaakwerk in uit de vorige sprint en alle taken gerelateerd aan vertalingen.

De aanpak voor de vertalingen werd uiteengezet in bijlage 9.11 VERTALINGEN. Om dit te realiseren was redelijk wat onderzoek vereist. De doorslag voor een aanpak kwam na enkele gesprekken met ontwikkelaars van telecomoperator Mobile Vikings. De operator biedt apps aan voor Android, iOS en Windows Phone. De apps worden op het moment van schrijven nog per platform apart ontwikkeld, maar delen wel hun vertalingen. Het zou zonde zijn om een applicatie die op elk mobiel platform exact dezelfde functies aanbiedt, meerdere keren te moeten vertalen. De schermen verschillen qua structuur vaak op elk platform, maar de knoppen om deze functies op te roepen zijn altijd wel op elk mobiel platform ergens in de applicatie terug te vinden.

Vervolgens moest bepaald worden hoe de taal van de applicatie ingesteld werd. Er werd geopteerd om automatisch de taal te kiezen van de smartphone waarop de app draait. De gebruiker heeft in de app wel de mogelijkheid om de taal te bepalen van de informatie die opgehaald en weergegeven wordt.

Ten slotte werden de laatste dagen van de sprint besteed aan het oplossen van enkele aanslepende bugs en het voorbereiden van de slotpresentatie. Op maandag 25 april werd een presentatie van het project en de volledige applicatie gegeven waarbij ook RealDolmen Education aanwezig was. Aan het einde van de laatste sprint bevatte de applicatie nog enkele bugs, maar aangezien die te wijten zijn aan afhankelijkheden zoals MVVM Light, konden ze niet meteen opgelost worden. In 9.6.1 MVVM LIGHT TOOLKIT zijn ze beschreven. In theorie zullen ze dan ook verdwijnen bij updates van de afhankelijkheden.

Later die week vond Xamarin Evolve plaats. Evolve is de jaarlijkse conferentie voor alle Xamarin ontwikkelaars waarin het bedrijf hun plannen voor het komende jaar uit de doeken doet. Gezien de rechtstreekse relevantie met de stage werd een verslag van dit evenement nog mee opgenomen.

## 5 Besluit

De stage is grotendeels verlopen zoals verwacht en voorzien in de voorbereidende fases uit het eerste semester. Aan het einde van de laatste sprint bevatte de applicatie alle gevraagde functies en was alle tussentijdse feedback mee opgenomen in de app.

Naast de gevraagde functionele en niet-functionele vereisten, werd een significante hoeveelheid tijd besteed aan nieuwe technische concepten, een solide architectuur en een degelijke manier van werken. Deze concepten en andere interessante aspecten zijn gedocumenteerd en als bijlage bij deze bachelorproef gevoegd.

Bij softwareontwikkeling is de eenvoudigste manier om iets te ontwikkelen niet altijd de beste. Op termijn moet software onderhoudbaar, betrouwbaar en kwalitatief blijven. Zonder uitgebreide voorbereiding en hulpmiddelen zoals unit tests en continue integratie is dit niet mogelijk. Bij dit project is dit zeker gebleken. Een softwareproject is nooit helemaal voltooid en ook deze applicatie kan zeker nog uitgebreid en verdere gedetailleerde afwerking gebruiken. Zonder stabiele basis is dit echter een hels karwei en dat moest dan ook vermeden worden. In het andere geval was de stage vermoedelijk veel minder interessant geweest en had het project een lagere leercurve aangeboden met overvloedig repetitieve taken.

Op technisch vlak werd daarnaast duidelijk dat cross-platform apps nog steeds een nieuw fenomeen is dat sterk onderhevig is aan verandering. Tijdens het project kregen heel wat afhankelijkheden, waaronder Xamarin zelf, meerdere updates met nieuwe mogelijkheden en oplossingen voor bekende bugs. Dat is ook merkbaar aan de documentatie. Voor Xamarin zelf bestaan ruim voldoende voorbeelden en documentatie, maar voor de meeste hulpmiddelen is het tegendeel waar.

Nu Xamarin onder de vleugels van Microsoft valt, is de toekomst van het product weer wat duidelijker. Het hoort perfect thuis in Microsofts verhaal om het .NET-platform naar elk toestel te brengen, ongeacht of het toestel verder met Microsoft Windows, Apple OS X of een ander besturingssysteem werkt. Xamarin vormt hier een uitgebreid verlengstuk om ontwikkelaars de kans te geven om één app te ontwikkelen en die met een minimum aan inspanningen te laten werken op elke smartphone, elke tablet, elke computer, elke televisie, elke smartwatch... De mogelijkheden blijven toenemen en het feit dat bij deze stage meer dan 70% van de code gedeeld kon worden op drie mobiele platformen bevestigt dat alleen maar.

Niet alleen het uitvoeren van één applicatie wordt eenvoudiger en cross-platform ondersteund. Ook de ontwikkeling ervan. Het is mogelijk geworden om C#-code te schrijven en te compileren op elk platform. Xamarin Studio voor Mac is hier een prachtig voorbeeld van. Vermoedelijk groeit dit op termijn uit tot Visual Studio voor Mac en kan dan elk type .NET-applicatie ook op Mac of Linux geschreven worden.

Naast het technische aspect, kwam bij de stage ook het professionele aspect kijken. RealDolmen werkt nagenoeg uitsluitend met consultants waardoor er dagelijks wel andere gezichten te bespeuren vallen. Ook voor RealDolmen is Xamarin nog heel nieuw, maar dankzij de vele ervaring met .NET kon men toch een interessante stage aanbieden met voldoende ondersteuning. Om het project tot een goed einde te brengen was dan ook samenwerking vereist met heel wat partijen en werd er regelmatig beroep gedaan op inzichten en ervaringen van collega's.

Hierbij kwamen zowel positieve als negatieve ervaringen bij kijken. Ook de typische kenmerken van een groot bedrijf kwamen naar boven. De hardware die nodig was om het project uit te voeren, was

niet tijdig voorzien of getest. Daarnaast kon het stagebedrijf wel het gemak bieden van meerdere kantoren met voldoende ruimte.

Ook de aangeleerde werkmethodes werden op de proef gesteld. De scrum-methode werd in de praktijk gebracht met zes sprints van elk twee weken en na elke sprint werd de aanpak en voortgang van het project geëvalueerd.

Buiten de omgeving van het stagebedrijf zelf vonden enkele interessante evenementen plaats waarbij nieuwe technologie uit de doeken gedaan werd en onderlinge kennisdeling met andere softwareontwikkelaars gefaciliteerd werd.

Er kan geconcludeerd worden dat zowel het project op zich als de stage succesvol waren en waardevolle vruchten hebben afgeworpen. Wat er verder met de ontwikkelde software gebeurt, valt buiten het bereik van dit project, maar de applicatie kan met een minimum aan werk omgezet worden in een sterk product van RealDolmen.

## 6 Zelfreflectie

Net geen decennium geleden kwamen de eerste smartphones op de markt en trokken ze meteen mijn aandacht. Die passie voor mobiele technologie is blijven groeien en resulteerde in deze stage rond de ontwikkeling van mobiele applicaties. Op vlak van softwareontwikkeling leunt mijn interesse vooral over naar Microsofts .NET-platform waardoor de ontwikkeling van mobiele applicaties in .NET een logische keuze geworden is.

De verscheidenheid aan mobiele platformen en types mobiele toestellen heeft een nieuwe trend ingezet: cross-platform applicaties. Dankzij een product zoals Xamarin kunnen ontwikkelaars grotendeels dezelfde code gebruiken om applicaties op meerdere mobiele platformen te laten werken. Spijtig genoeg was dit product in het verleden<sup>2</sup> zeer duur en waren de kansen om ermee aan de slag te gaan dus vrij beperkt. Een stage is dus een uitgelezen moment om deze kans toch nog te grijpen.

Daarom ben ik ook zeer tevreden over de kansen die de stage mij geboden heeft. De enorme hoeveelheid nieuwe ervaringen, technische inzichten en de mogelijkheid om een echt project te kunnen opleveren, waren zeer waardevol. Het was eveneens mogelijk om een eenvoudig project af te leveren dat voldeed aan de wensen van de klant, maar weinig kwaliteit bood. Dit heb ik bewust vermeden en waar mogelijk heb ik de kans gegrepen om aspecten dieper uit te spitten en met oog voor detail af te werken.

De stage was overigens niet altijd rozengeur en maneschijn. Enkele afhankelijkheden bleken achteraf een slechte keuze te zijn en zorgden voor bugs. Ook op vlak van technische ondersteuning botste ik soms op beperkingen. Hier had een doordachtere keuze voor de afhankelijkheden waarschijnlijk verstandiger geweest.

Naast de specifieke kennis rond Xamarin, kwamen enkele competenties uit de opleiding Toegepaste Informatica aan bod. Deze werden gedocumenteerd in bijlage 9.4 POP / COMPETENTIES.

Als softwareontwikkelaar ben ik ook gegroeid op vlak van aanpak. Het is veel duidelijker geworden hoe het er op de werkvloer in een groot bedrijf als RealDolmen aan toe gaat en hoe professioneel wordt samengewerkt met alle stakeholders van een project.

Kortom zorgde de stage voor een goede leercurve dankzij de zowel de nieuwigheden rond cross-platform applicaties als het uitdiepen van enkele technische aspecten en voor heel waardevolle ervaringen in een professionele werkomgeving. Het is dan ook zeer deugddoend om een succesvol eindproduct te kunnen afleveren als onderdeel van het stageproject.

---

<sup>2</sup> Ondertussen is Xamarin volledig gratis geworden. Dit gebeurde na de overname door Microsoft die tijdens de stage plaatsvond.

## 7 Figuren en tabellen

Figuur 1: logo van RealDolmen .....	8
Figuur 2: de website van RealDolmen Education .....	10
Figuur 3: burndown chart sprint 1 .....	14
Figuur 4: burndown chart sprint 2 .....	16
Figuur 5: burndown chart sprint 3 .....	18
Figuur 6: burndown chart sprint 4 .....	19
Figuur 7: burndown chart sprint 5 .....	21
Figuur 8: burndown chart sprint 6 .....	24
Figuur 9: sprint 1 .....	39
Figuur 10: sprint 2 (week 1) .....	39
Figuur 11: burndown chart week 4 .....	42
Figuur 12: unit test coverage bij view models ligt boven 80% .....	45
Figuur 13: sprint 2 .....	45
Figuur 14: sprint 3 .....	46
Figuur 15: verduidelijking van de werking van caching .....	47
Figuur 16: burndown chart week 6 .....	48
Figuur 17: reeds afgewerkte delen .....	52
Figuur 18: af te werken delen .....	52
Figuur 19: verhouding van Rx tot andere componenten in .NET .....	60
Figuur 20: aantal lijnen code per project .....	71
Figuur 21: overzicht van het project met de gedeelde code .....	73
Figuur 22: statische gegevens (geen business logica) in de gedeelde code .....	74
Figuur 23: business logica in de gedeelde code (view models en services) .....	74
Figuur 24: view models met hun respectievelijke unit tests .....	75
Figuur 25: overzicht van de code van de Android app .....	75
Figuur 26: overzicht van de code van de iOS app .....	76
Figuur 27: overzicht van de code van de Windows Phone app .....	76
Figuur 28: platformspecifieke services .....	77
Figuur 29: inhoud van Model bij MVVM .....	78
Figuur 30: inhoud van View bij MVVM .....	79
Figuur 31: ViewModel-component bij MVVM .....	79
Figuur 32: voorbeeld van dependency injection (Web Camps Team, 2013) .....	81
Figuur 33: werking van observable-observer patroon .....	81
Figuur 34: gegevensmodel van RealDolmen Education .....	83
Figuur 35: gebruik van cache in de eerste fases van de applicatie .....	84
Figuur 36: verloop van authenticatie en verhouding gedeelde code .....	86
Figuur 37: hoe code gedeeld wordt bij authenticatie .....	87
Figuur 38: vereenvoudigd verloop van implicit flow .....	88
Figuur 39: dashboard van Xamarin.Insights .....	91
Figuur 40: voorbeeld van exception op Xamarin.Insights .....	91
Figuur 41: implementaties voor vertalingen .....	94
Figuur 42: overzicht van POEditor .....	94
Figuur 43: test coverage .....	96
Figuur 44: screenshot van hoofdscherm in de Android app .....	97
Figuur 45: screenshot van hoofdscherm in de iOS app .....	97
Figuur 46: screenshot van hoofdscherm in de Windows Phone app .....	97

Figuur 47: screenshot van scherm om feedback te geven op Windows Phone.....	97
Figuur 48: screenshot van inschrijfscherm op Windows Phone.....	97
Figuur 49: kleine tegel op Windows Phone.....	98
Figuur 50: brede tegel op Windows Phone .....	98
Figuur 51: blauwe statusbalk op Windows Phone .....	98
Figuur 52: vlakke elementen op Windows Phone .....	98
Figuur 53: voorbeeld van Material Design in de Android app .....	98
Figuur 54: voorbeeld van Material Design in de Android app .....	98
Figuur 55: voorbeeld van Material Design in de Android app .....	98
Figuur 56: lifecycle van een activity (Google Inc.) .....	99
Figuur 57: lifecycle van een fragment (Google Inc.).....	99
Figuur 58: voorbeeld van vlak design op iOS .....	100
Figuur 59: voorbeeld van vlak design op iOS .....	100
Figuur 60: voorbeeld van vlak design op iOS .....	100
Figuur 61: ontwikkelingscyclus bij gebruik van VCS en CI.....	103
Figuur 62: voorbeeld van de AnimatedCircleLoadingView component.....	104
 Tabel 1: planning sprint 1 .....	13
Tabel 2: planning sprint 2 .....	15
Tabel 3: planning sprint 3 .....	17
Tabel 4: planning sprint 4 .....	19
Tabel 5: planning sprint 5 .....	21
Tabel 6: planning sprint 6.....	23
Tabel 7: werkuren .....	34
Tabel 8: tijdsbesteding evenementen .....	35
Tabel 9: tijdsbesteding documentatie.....	36
Tabel 10: overzicht van de scores voor de competenties in het POP .....	68
Tabel 11: structuur van de applicatie(s) .....	71
Tabel 12: hoeveelheid gedeelde code .....	71
Tabel 13: ondersteunde ontwikkelconfiguraties (groen: beste ontwikkelervaring / meeste ondersteuning).....	101

## 8 Bibliografie

- (1995). In E. Gamma, R. Helm, R. Johnson, & J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software* (pp. 97-117,127-135,233-243,293-305). United States of America: Addison-Wesley.
- Cabus, W. (2016, april 16). *GAB Huizingen*. Opgehaald van Global Azure Bootcamp: <http://huizingen-be.azurebootcamp.net/>
- Fielding, R. T. (2000). Representational State Transfer (REST). *Architectural Styles and the Design of Network-based Software Architectures*. Irvine: University of California. Opgehaald van [http://www.ics.uci.edu/~fielding/pubs/dissertation/rest\\_arch\\_style.htm](http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm)
- Gibbens, R. (2015, januari 13). *Resilient network services with mobile Xamarin apps*. Opgehaald van Artek Software: <http://arteksoftware.com/resilient-network-services-with-xamarin/>
- Google Inc. (sd). *Fragments*. Opgehaald van Android Developers: <https://developer.android.com/guide/components/fragments.html>
- Google Inc. (sd). *Managing the Activity Lifecycle*. Opgehaald van Android Developers: <https://developer.android.com/training/basics/activity-lifecycle/index.html>
- MADN. (2016, februari 23). *Git and Beyond*. Opgehaald van Evenbrite: <http://www.eventbrite.com/e/git-and-beyond-tickets-21458263243>
- Martin, R. C. (2000). *Principles of Object-Oriented Design*. Opgehaald van Uncle Bob: <http://butunclebob.com/ArticleS.UncleBob.PrinciplesOfOOD>
- Microsoft Belux. (2016, april 5). *Beyond Xamarin 101*. Opgehaald van Meetup: <http://www.meetup.com/Belgian-Mobile-NET-Developers-Group/events/229412338/>
- Microsoft Corp. (2007). *The Service Locator Pattern*. Opgehaald van Microsoft Developer Network: <https://msdn.microsoft.com/en-us/library/ff648968.aspx>
- Microsoft Corp. (2016). Build. *Build*. The Moscone Center: Microsoft Corp. Opgeroepen op maart 30, 2016
- Microsoft Developer Network. (2012, februari 10). *The MVVM Pattern*. Opgehaald van Microsoft Developer Network: <https://msdn.microsoft.com/en-us/library/hh848246.aspx>
- Mono Project. (sd). *Home*. Opgehaald van Mono: <http://www.mono-project.com/>
- Nabar, S. (2015, november 23). *Demystifying HttpClient APIs in the Universal Windows Platform*. Opgehaald van Building Apps For Windows: <https://blogs.windows.com/buildingapps/2015/11/23/demystifying-httpclient-apis-in-the-universal-windows-platform/>
- Odisee UC. (2015, september). *Competentieprofielen / Opleidingsspecifieke leerresultaten (OLR) academiejaar 2015-2016*. Opgehaald van Odisee: <https://webapps.odisee.be/ECTSCompetenties/Competentieprofiel.aspx?taal=N&OPLID=51&ACJ=2015>
- RealDolmen NV. (sd). *Opleidingen*. Opgehaald van RealDolmen: <http://www.realdolmen.com/nl/education/about-us>

Vermeir, N. (2016, maart 4). *Git and Beyond*. Opgehaald van MADN: <http://madn.be/post/2016/03/04/Slides-for-Git-and-Beyond.aspx>

VISUG. (2016, maart 30). *Visug and MADN present you The Build Keynote 2016*. Opgehaald van Visug.be: <http://www.visug.be/PassedEvents/Details/27>

Web Camps Team. (2013, februari 18). *ASP.NET MVC 4 Dependency Injection*. Opgehaald van ASP.NET: <http://www.asp.net/mvc/overview/older-versions/hands-on-labs/aspnet-mvc-4-dependency-injection>

Xamarin Inc. (2016, april 27). *Xamarin Evolve Live*. Opgehaald van Xamarin Evolve: <https://evolve.xamarin.com/live>

## 9 Bijlagen

### 9.1 Tijdsbesteding

De tijdsbesteding van deze stage dient in context bekeken te worden met de tijdsbesteding van de bachelorproef. Er ging namelijk heel wat werk aan de stage vooraf dat ook bij de bachelorproef ingedeeld kan worden.

## 9.1.1 Werkuren

Week	Datum	Begin	Einde	Pauze	Totaal
6	ma 1 feb	08h50	17h40	00h30	08h20
6	di 2 feb	08h35	17h55	00h30	08h50
6	wo 3 feb	07h45	17h20	00h30	09h05
6	do 4 feb	08h50	17h40	00h40	08h10
6	vr 5 feb	08h55	15h25	00h45	05h45
Totaal 6					40h10
7	ma 8 feb	08h40	17h10	00h30	08h00
7	di 9 feb	09h20	18h20	00h30	08h30
7	wo 10 feb	08h55	17h20	00h30	07h55
7	do 11 feb	ziek			00h00
7	vr 12 feb	07h55	18h30	00h30	10h05
Totaal 7					34h30
8	ma 15 feb	08h05	17h10	00h30	08h35
8	di 16 feb	08h45	17h05	00h30	07h50
8	wo 17 feb	09h10	18h15	00h40	08h25
8	do 18 feb	08h55	17h20	00h30	07h55
8	vr 19 feb	07h55	15h30	00h30	07h05
Totaal 8					39h50
9	ma 22 feb	08h05	18h25	00h30	09h50
9	di 23 feb	09h55	16h55	00h30	06h30
9	wo 24 feb	08h45	17h10	00h30	07h55
9	do 25 feb	09h15	18h45	00h30	09h00
9	vr 26 feb	08h00	15h20	00h30	06h50
Totaal 9					40h05
10	ma 29 feb	09h00	16h40	00h30	07h10
10	di 1 mrt	08h55	17h30	00h30	08h05
10	wo 2 mrt	09h00	16h10	00h30	06h40
10	do 3 mrt	09h00	19h05	00h30	09h35
10	vr 4 mrt	08h00	17h00	00h30	08h30
Totaal 10					40h00
11	ma 7 mrt	09h00	16h55	00h30	07h25
11	di 8 mrt	07h20	16h45	00h30	08h55
11	wo 9 mrt	09h40	17h40	00h30	07h30
11	do 10 mrt	aanwezig op Odisee			00h00
11	vr 11 mrt	08h20	17h05	00h30	08h15
Totaal 11					32h05
12	ma 14 mrt	09h55	17h00	00h30	06h35
12	di 15 mrt	09h25	16h55	00h30	07h00
12	wo 16 mrt	07h55	16h40	00h30	08h15
12	do 17 mrt	09h35	19h15	00h30	09h10
12	vr 18 mrt	08h15	17h55	00h30	09h10
Totaal 12					40h10
13	ma 21 mrt	07h50	17h40	00h30	09h20
13	di 22 mrt	10h30	19h00	00h30	08h00

13	wo 23 mrt	09h30	18h50	00h30	08h50
13	do 24 mrt	09h35	19h05	00h30	09h00
13	vr 25 mrt	07h50	13h25	00h30	05h05
Totaal 13					40h15
14	di 29 mrt	08h00	18h50	00h30	10h20
14	wo 30 mrt	09h45	16h55	00h30	06h40
14	do 31 mrt	10h55	17h15	00h30	05h50
14	vr 1 apr	08h00	17h45	00h30	09h15
Totaal 14					32h05
15	ma 4 apr	09h50	17h25	00h55	06h40
15	di 5 apr	10h30	17h55	00h40	06h45
15	wo 6 apr	09h45	16h10	00h30	05h55
15	do 7 apr	08h35	19h40	00h30	10h35
15	vr 8 apr	08h00	18h40	00h30	10h10
Totaal 15					40h05
16	ma 11 apr	09h50	18h45	00h45	08h10
16	di 12 apr	09h50	16h50	00h30	06h30
16	wo 13 apr	09h15	18h10	00h30	08h25
16	do 14 apr	10h10	19h00	00h30	08h20
16	vr 15 apr	08h00	17h10	00h30	08h40
Totaal 16					40h05
17	ma 18 apr	08h55	17h15	00h30	07h50
17	di 19 apr	09h50	19h10	00h30	08h50
17	wo 20 apr	10h10	17h05	00h30	06h25
17	do 21 apr	10h05	19h30	00h30	08h55
17	vr 22 apr	08h00	16h30	00h30	08h00
Totaal 17					40h00
18	ma 25 apr	08h10	16h40	00h30	08h00
Totaal 18					08h00
<b>Eindtotaal</b>					<b>467h20</b>

Tabel 7: werkuren

### 9.1.2 Evenementen

Evenement	Datum	Begin	Einde	Duur
<b>GIT and Beyond</b>	23/02/2016	17:30	22:00	4h 30m
<b>Afspraak stagebegeleiding</b>	4/03/2016	18:00	19:00	1h 0m
<b>Jobbeurs</b>	10/03/2016	8:30	16:00	7h 30m
<b>Build Keynote</b>	30/03/2016	17:00	21:30	4h 30m
<b>Build dag 2</b>	31/03/2016	17:30	20:00	2h 30m
<b>Build dag 3</b>	1/04/2016	17:30	20:00	2h 30m
<b>Beyond Xamarin 101</b>	5/04/2016	18:00	22:30	4h 30m
<b>Global Azure Bootcamp</b>	16/04/2016	8:30	17:00	8h 30m
<b>Afspraak stagebegeleiding</b>	22/04/2016	17:10	17:50	0h 40m
<b>Xamarin Evolve dag 1</b>	27/04/2016	15:00	18:00	3h 0m
<b>Xamarin Evolve dag 2</b>	28/04/2016	15:00	23:30	8h 30m
<b>Totaal</b>				<b>47h 40m</b>

Tabel 8: tijdsbesteding evenementen

## 9.1.3 Opstellen documentatie

Onderwerp	Aantal	Duur per document	Totale duur
Integratietekst	1	80hoo	80hoo
Weekverslag	12	01hoo	12hoo
Presentatie stage Odisee	1	10hoo	10hoo
Evenementverslag	5	02hoo	10hoo
Presentatie stage RealDolmen	1	06hoo	06hoo
Overige (beheer Google Drive, e-mailverkeer...)	1	03hoo	03hoo
Presentatie MS Community	1	02hoo	02hoo
<b>Totaal</b>			<b>123hoo</b>

Tabel 9: tijdsbesteding documentatie

## 9.2 Weekverslagen

### 9.2.1 Week 1

De week begon rustig met een introductie van de infrastructuur en gewoontes bij RealDolmen. Maandag heb ik mijn badge en laptop opgehaald, de nodige software<sup>3</sup> geïnstalleerd en de product backlog opgesteld. De installatie van de software duurde vier uren en de product backlog<sup>4</sup> opstellen heeft een tweetal uren gekost. Vervolgens ben ik aan de slag gegaan met het opzetten van het project en de architectuur<sup>5</sup> van het project.

Dinsdag begon moeilijker. Maandagnamiddag had ik vastgesteld dat voor verschillende software zoals de Android SDK toegang nodig is tot het minder beveiligde (draadloze) netwerk van RealDolmen omdat die software anders geen verbinding met het internet kan maken. De laptop kon hier niet mee verbinden door driver-problemen. Er was ook te weinig RAM-geheugen beschikbaar om een simulator van Android of Windows Phone te draaien. Daarom besloot de interne helpdesk dat het gemakkelijker ging zijn om het geheugen uit te breiden en Windows 10 op de laptop te installeren aangezien de drivers voor Windows 10 wel goed zouden werken. Door de installatie hiervan moest ik nadat wel alle software weer opnieuw installeren. Daarom besloot ik dit uit te stellen tot woensdag en woensdag mijn eigen laptop mee te nemen om op te werken tijdens de herinstallatie.

In de loop van de dag heb ik dinsdag gewerkt aan het splash screen voor de app op Android en op Windows Phone. Ook heb ik verschillende online lessen gevolgd en artikels gelezen over een doordachte structuur voor cross-platform apps. Ik heb gekozen voor MVVM op basis van de MVVM Light toolkit. Hiermee kan er zoveel mogelijk code gedeeld worden over de verschillende mobiele platformen en is de structuur doorheen de app heel overzichtelijk. Dankzij een dependency injection framework zoals Autofac was ik vervolgens in staat om heel duidelijk alle code in te delen.

Woensdag nam de herinstallatie van de laptop veel tijd in beslag en begon ik aan de code om categorieën van opleidingen op te halen. Ik had wat problemen met de weergave van code in design-time. Hiermee kan ik de app visueel ontwerpen met testgegevens in Blend en Visual Studio. Dit lukte niet meteen door een bug in mijn code rond dependency injection.

Donderdag en vrijdag heb ik verder gewerkt aan de code om categorieën op te halen en deze vervolgens weer te geven op Android en Windows Phone. Ook de huisstijl van RealDolmen werd hierin geïntegreerd en er werd van in het begin al nagedacht over hoe de app op de eenvoudigste manier kan vertaald worden naar de drie gewenste talen zonder alle vertalingen per mobiel platform opnieuw te moeten doen. Xamarin.Insights werd geïntegreerd om fouten, logs en crash reports van de app te kunnen verzamelen in een online dashboard.

In het weekend heb ik het bewijsmateriaal verzameld.

---

<sup>3</sup> Visual Studio, Resharper, Xamarin en heel wat plugins om het werken hiermee te vereenvoudigen

<sup>4</sup> De backlog is te vinden in de map 'Management' onder 'Bewijsmateriaal'.

<sup>5</sup> De architectuur wordt weergegeven in de bijlage 'Architectuur' in de map 'Product' onder 'Bewijsmateriaal'.

### 9.2.2 Week 2

Deze week lag de focus op de het opzetten van de MacBook voor de ontwikkeling van de iOS-app en het opzetten van continuous integration. Donderdag was ik ziek, waardoor ik ook heel wat tijd verloren heb.

De volledige product backlog en alle taken worden bijgehouden op Visual Studio online. Ook alle broncode wordt hier opgeslagen onder verschillende check-ins. Elke check-in omvat een stuk code dat een enkel doel heeft. Het zou dan ook zinvol zijn om bij elke check-in na te gaan of de code wel nog gecompileerd kan worden en of alle testen slagen. Hiervoor dient continuous integration (CI).

Met de hulp van Nico Vermeir die mij de nodige rechten moest verlenen, heb ik een omgeving opgezet waarbij alle broncode van de Android app, de Windows Phone app en de gedeelde code gecompileerd en getest wordt na elke check-in. Voor de iOS-app is een MacBook nodig en dit wordt dan ook volgende week opgezet. Om de Android app te kunnen compileren had Visual Studio online een licentie nodig voor Xamarin die ik dan ook heb geconfigureerd.

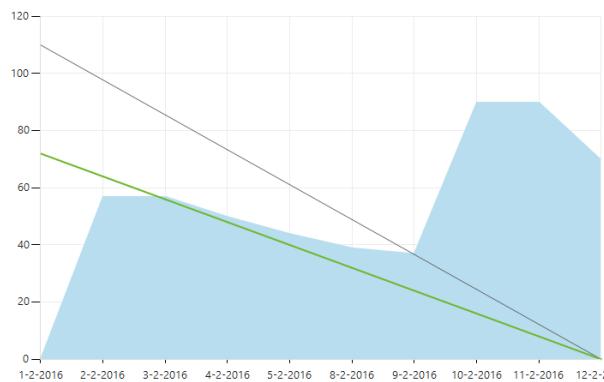
Maandag gebeurde ook de afwerking van de code van de week ervoor. Aan de hand van enkele handige NuGet packages (Gibbens, 2015) werd de code meteen veel ordelijker en performanter. Zo werd alle boilerplate code om verbinding te maken met de API geabstraheerd en werd caching geïmplementeerd. De categorieën laden veel sneller en de app werkt ook wanneer de smartphone van de gebruiker offline is.

Tenslotte werd aan de authenticatie gesleuteld. Die kan niet op elk platform op dezelfde manier verlopen. De theorie erachter is wel dezelfde, maar elk mobiel platform gaat anders om met het openen van een browser om de gebruiker te laten inloggen. Na het inloggen kan de app een token (soort van code) opvragen. Met dit token kunnen authenticatiegegevens van de gebruiker opgehaald worden. Momenteel werkt dit systeem nog niet omdat de STS (Security Token Service) nog niet juist geconfigureerd is. Dit valt buiten de scope van de stage en moet dus door RealDolmen gebeuren.

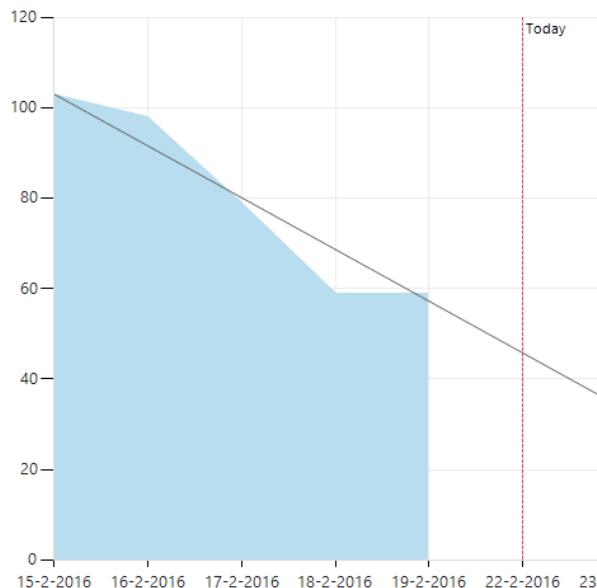
### 9.2.3 Week 3

De tijd is voorbijgevlogen en maandag was het al tijd voor de eerste sprint review. Ik stelde de voortgang van het project voor aan Nico Vermeir (stagebegeleider RD), Tom Knockaert (manager RD Education), James Dejaeghere (.NET developer) en Alexander Reynaert (.NET developer). De iOS app stelde nog niets meer voor dan een icoontje en door ziekte (donderdag) en enkele lastige bugs en laptopproblemen waren de Android en Windows Phone apps ook achter op schema geraakt.

Bij het opstellen van de sprint backlog voor sprint 2 werden de onafgewerkte taken uit sprint 1 dan ook overgedragen. Op het einde van week 1 van sprint 2 is de achterstand volledig weggewerkt.



Figuur 9: sprint 1



Figuur 10: sprint 2 (week 1)

De app staat nu dan ook veel verder. Authenticatie lukt nog niet helemaal, maar zit al in een fase verder. Om dit af te werken is de aanwezigheid van mijn stagebegeleider vereist (enkel maandagvoormiddag). iOS development verliep vrij moeilijk in het begin, vooral vanuit Visual Studio op Windows. De ontwikkeling zelf verliep heel traag, dus ben ik overgestapt op Xamarin Studio op de MacBook. Daarop werkt het visueel ontwerpen van de iOS app veel sneller en stabiever.

iOS bevat geen mogelijkheid om een centraal thema in te stellen zoals Android en Windows Phone. Enkele eigenschappen kunnen globaal gewijzigd worden, maar dit is slechts een minderheid. De ontwikkeling verloopt ook heel anders. Visueel design gebeurt ofwel in C#, ofwel met een GUI die een met Storyboards kan werken. Nadat je doorhebt hoe het in elkaar zit, is het wel een prettige manier van werken.

Na authenticatie ging ik aan de slag met de zoekfunctie en met de subcategorieën. De API-methodes voor de zoekfunctie zijn af en op Windows Phone werkt de zoekbalk al. Op Android en iOS staat de zoekbalk er al, maar is hij nog niet gekoppeld aan een actie.

Subcategorieën openen zit ongeveer in hetzelfde stadium. De API-methodes zijn af, maar er zijn nog geen visuele acties waarmee ze opgeroepen kunnen worden.

Daarnaast heb ik de TFS-repository omgevormd naar een GIT-repository zodat de code op een praktische manier gedeeld kan worden met de MacBook. Ook heb ik de build server geïnstalleerd. Alle code die ge-commiteert wordt, wordt automatisch gecompileerd in de cloud van Visual Studio Online zodat fouten in de code snel ontdekt kunnen worden. Ook alle beschikbare unit tests worden daar nog eens gedraaid.

Tot slot heb ik geprobeerd om het aantal unit testen verder uit te breiden, maar dit is niet gelukt omdat unit tests van de app-projecten niet automatisch uitgevoerd kunnen worden, buildfouten veroorzaken en er weinig tot geen documentatie over beschikbaar is.

## 9.2.4 Week 4

In de 2<sup>de</sup> week van sprint 2 lag de focus op de afwerking van alle user stories voor de eerste twee sprints. Dit is grotendeels wel gelukt. Er waren 80 fysieke uren beschikbaar, maar door achterstand uit de eerste sprint, waren er ca. 100 uren gepland. Op het einde van de sprint was er minder dan 20h achterstallig werk.

### 9.2.4.1 User stories

#### 9.2.4.1.1 Splash screen

Voltooid op Android, iOS en Windows Phone. Gedurende het vertonen van het splash screen, initialiseert de app op de achtergrond al enkel bronnen die nodig zijn in een latere fase van de app.

#### 9.2.4.1.2 Alle categorieën ophalen

Volledig voltooid op alle platformen.

#### 9.2.4.1.3 Authenticatie

Hier zit ik bij de drie platformen even ver, maar er is nog werk aan de winkel. De authenticatie op zich lukt al, maar de authenticatiegegevens worden nog nergens opgeslagen en het is voor de gebruiker nog niet duidelijk dat de authenticatie gelukt is.

#### 9.2.4.1.4 Zoeken naar een opleiding

Dit lukt nog niet op iOS, maar wel al op de andere platformen. Het enige dat nog ontbreekt op iOS is een oplossing voor een bug waardoor de invoer van het zoekveld niet gekoppeld kan worden aan een effectieve zoekopdracht.

#### 9.2.4.1.5 Lijst van subcategorieën na klikken op een categorie

Dit lukt volledig op alle platformen, maar op iOS geeft de knop om terug te keren naar het vorige scherm nog een crash zonder verdere omschrijving.

### 9.2.4.2 Bijkomende functies en taken

#### 9.2.4.2.1 Continuous integration

Na wat onderzoek ben ik erin geslaagd om continuous integration op te zetten voor alle code. Bij het comitten van nieuwe code wordt die onmiddellijk gecompileerd en worden alle unit tests gedraaid om er zeker van te zijn dat de nieuwe code geen problemen introduceert.

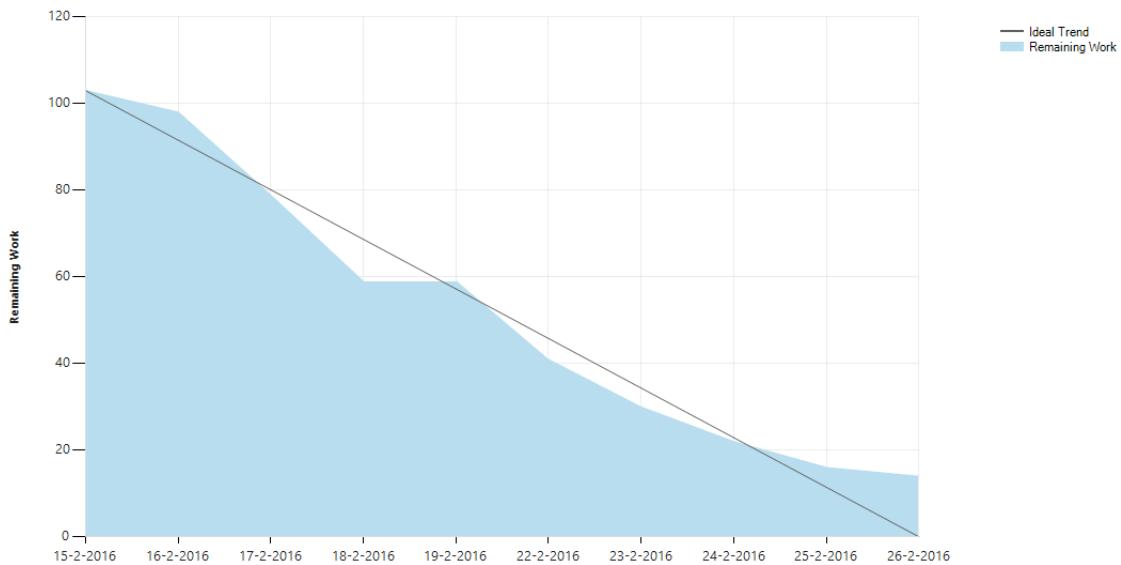
#### 9.2.4.2.2 Unit tests

Alle view models (business logic) worden volledig getest met unit tests. Hiervoor waren een 20-tal tests nodig. Zo is het zeker dat de view models naar behoren werken en niet de oorzaak kunnen zijn van problemen.

#### 9.2.4.2.3 Vertalingen

Het proces om vertalingen in te bouwen over de verschillende platformen heen werd grotendeels geautomatiseerd en de app is al volledig beschikbaar in het Nederlands en het Engels op Android en Windows Phone. Op iOS vereist dit bijkomstig werk dat weggelegd is voor een latere user story.

#### 9.2.4.3 Burndown chart



Figuur 11: burndown chart week 4

Zoals te zien is op bovenstaande burndown chart, werden de laatste dagen besteed weinig taken opgelost. Deze tijd werd namelijk besteed aan het oplossen van bugs.

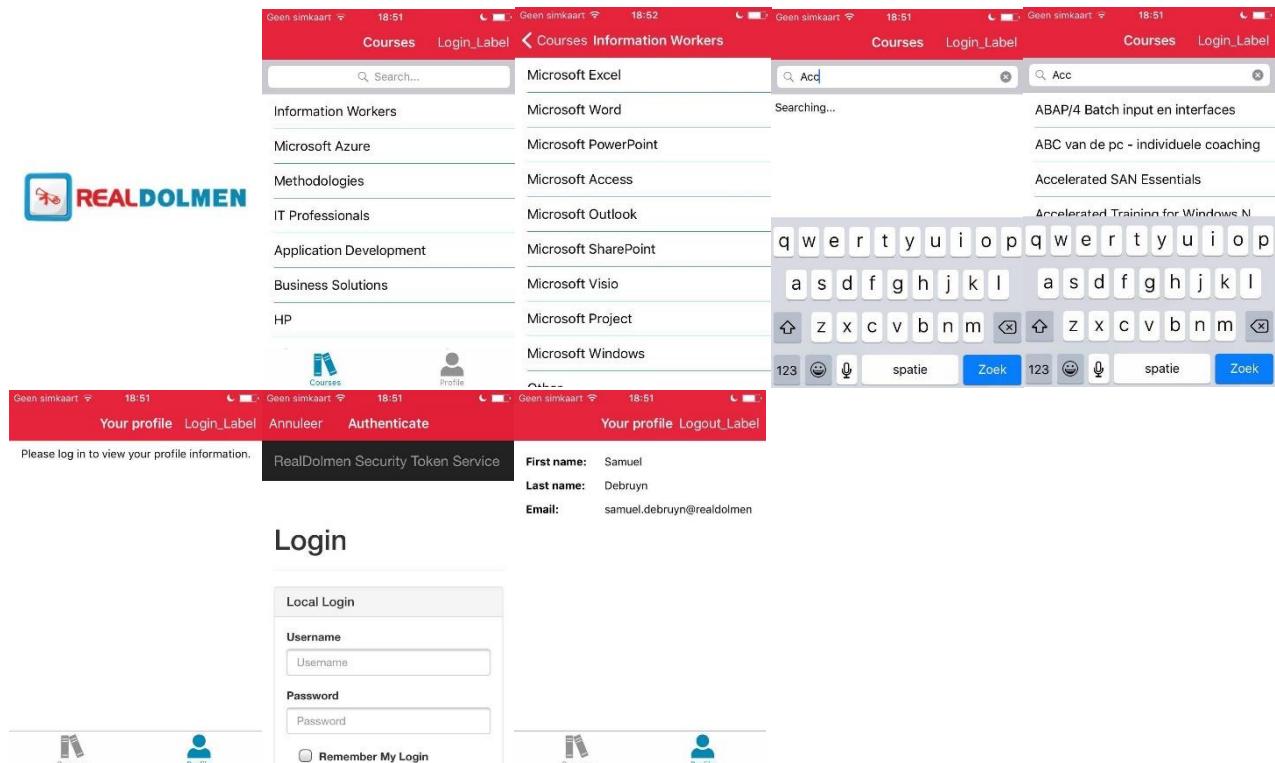
#### 9.2.4.4 Moeilijkheden bij de ontwikkeling

Bij crashes geeft de iOS app meestal geen foutmeldingen of exceptions. Dit vermoeilijkt het opsporen en verhelpen van bugs aanzienlijk. Op Android merk ik regelmatig hetzelfde op, maar de Android-versie werkt toch veel stabiever. Op Windows Phone doet dit probleem zich niet voor. Dit valt te verklaren door de achterliggende architectuur die ik beschreef in mijn bachelorproef

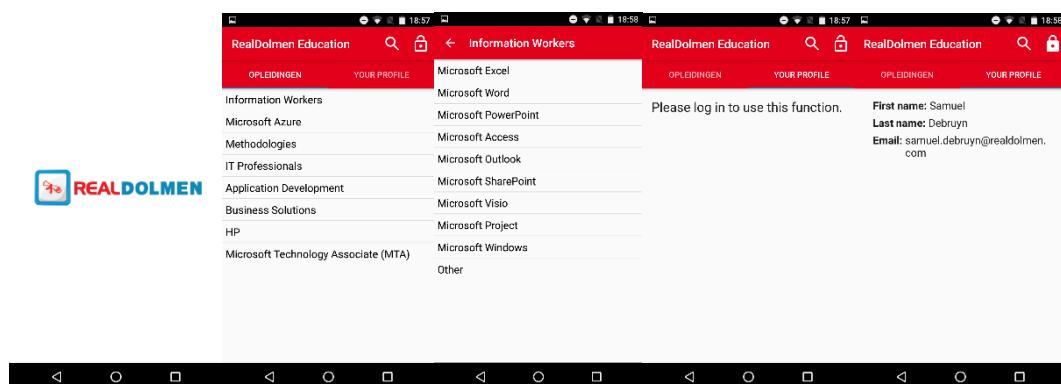
## 9.2.5 Week 5

### 9.2.5.1 Voorlopige screenshots

#### 9.2.5.1.1 iOS



#### 9.2.5.1.2 Android



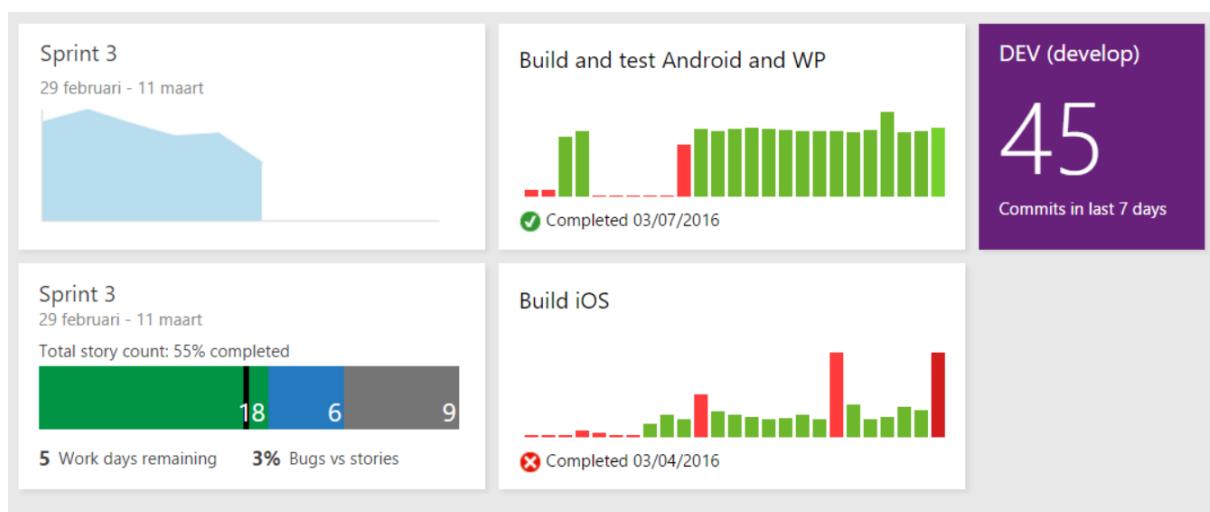
### 9.2.5.1.3 Windows Phone



### 9.2.5.2 Visual Studio Online

De voortgang van het project kan volledig online opgevolgd worden via  
[https://rdmscc.visualstudio.com/DefaultCollection/Stageproject%20Education%20Samuel/Stageproject%20Education%20Samuel%20Team/\\_dashboards](https://rdmscc.visualstudio.com/DefaultCollection/Stageproject%20Education%20Samuel/Stageproject%20Education%20Samuel%20Team/_dashboards)

Daar heb ik een dashboard gecreëerd met de burndown chart van de huidige sprint, de voortgang van de sprint, de status van de automatische builds van de app en het aantal code commits van de laatste 7 dagen.



De verklaring voor de mislukte builds ligt bij het niet beschikbaar zijn van toestellen waarop de builds uitgevoerd kunnen worden.

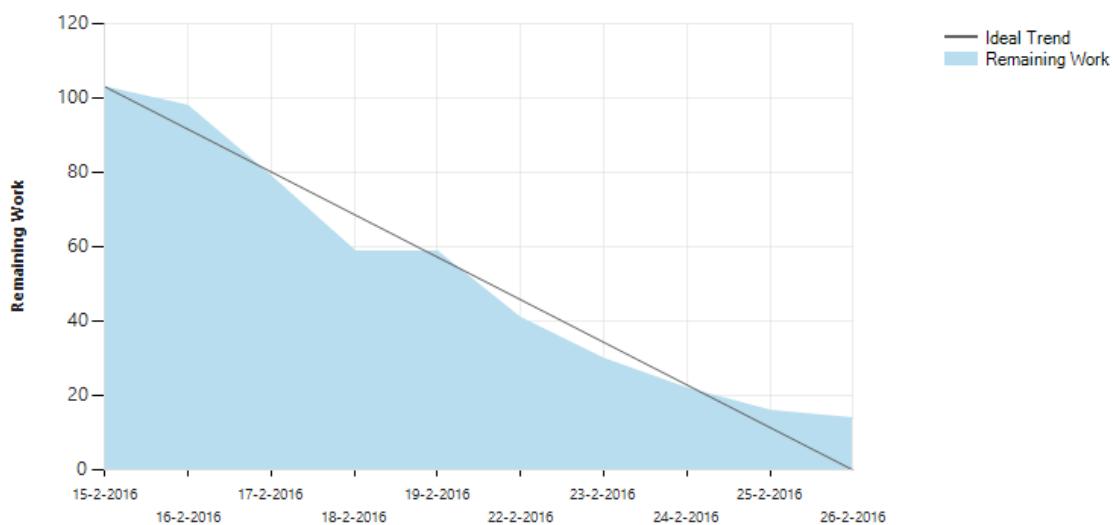
### 9.2.5.3 Unit tests

Sinds sprint 3 wordt de code van view models (alle business logica) getest via unit tests. Via tools zoals ReSharper kan de code coverage gemeten worden (hoeveelheid die getest wordt in verhouding met de totale hoeveelheid code).

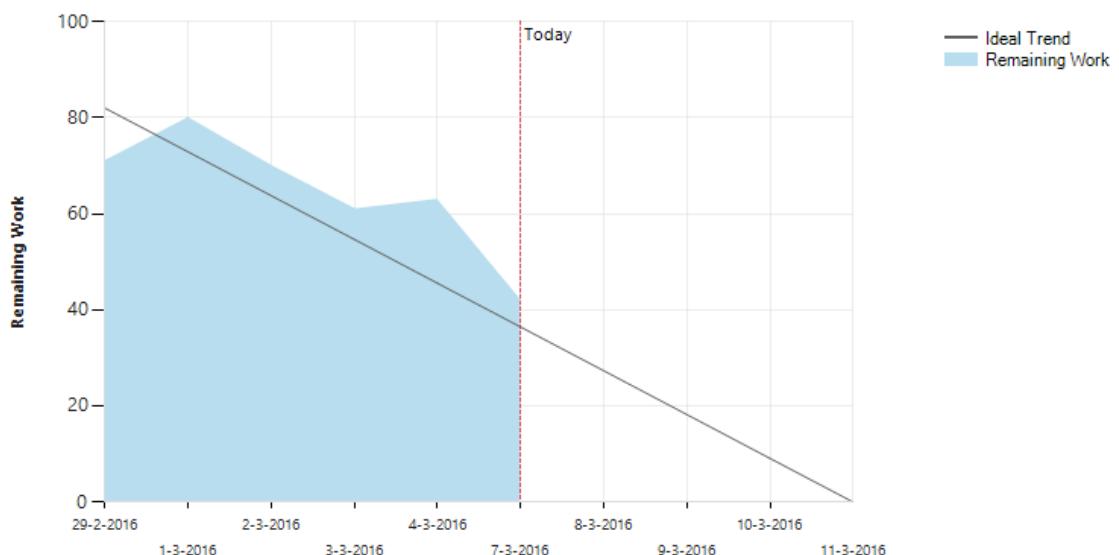
Symbol	Coverage (%) ▲	Uncovered/Total Stmt.
▲ ☐ Total	24%	2909/3821
▷ ☐ EducationApp.iOS	0%	436/436
▷ ☐ EducationApp.Droid	0%	453/453
▷ ☐ EducationApp.WinPhone	0%	1276/1276
▲ ☐ EducationApp	41%	742/1258
▷ ⟨⟩ EducationApp	0%	2/2
▷ ⟨⟩ EducationApp.Services.Web	0%	38/38
▷ ⟨⟩ EducationApp.Services.Web.Utilities	0%	67/67
▷ ⟨⟩ EducationApp.Services	4%	103/107
▷ ⟨⟩ EducationApp.Models	30%	417/598
▷ ⟨⟩ EducationApp.Exceptions	32%	13/19
▷ ⟨⟩ EducationApp.Services.Fakes	44%	53/94
▷ ⟨⟩ EducationApp.Extensions	72%	7/25
▲ ⟨⟩ EducationApp.ViewModels	85%	37/250
▷ ⚡ ContactViewModel	69%	23/74
▷ ⚡ CategoryViewModel	92%	5/60
▷ ⚡ MainViewModel	92%	9/116
▷ ⟨⟩ EducationApp.ViewModels.Utilities	90%	5/49
▷ ⟨⟩ EducationApp.Services.Fakes.Utilities	100%	0/4
▷ ⟨⟩ EducationApp.Messaging	100%	0/5
▷ ☐ EducationApp.UnitTests	99%	2/398

Figuur 12: unit test coverage bij view models ligt boven 80%

#### 9.2.5.4 Burndown charts sprints 2 en 3



Figuur 13: sprint 2



Figuur 14: sprint 3

De taken die overgedragen waren uit sprint 1, waren in sprint 2 volledig afgewerkt. Op het einde zat ik met een lichte achterstand door het afwerken van enkele bugs en de achterstand die bij sprint 1 was opgelopen.

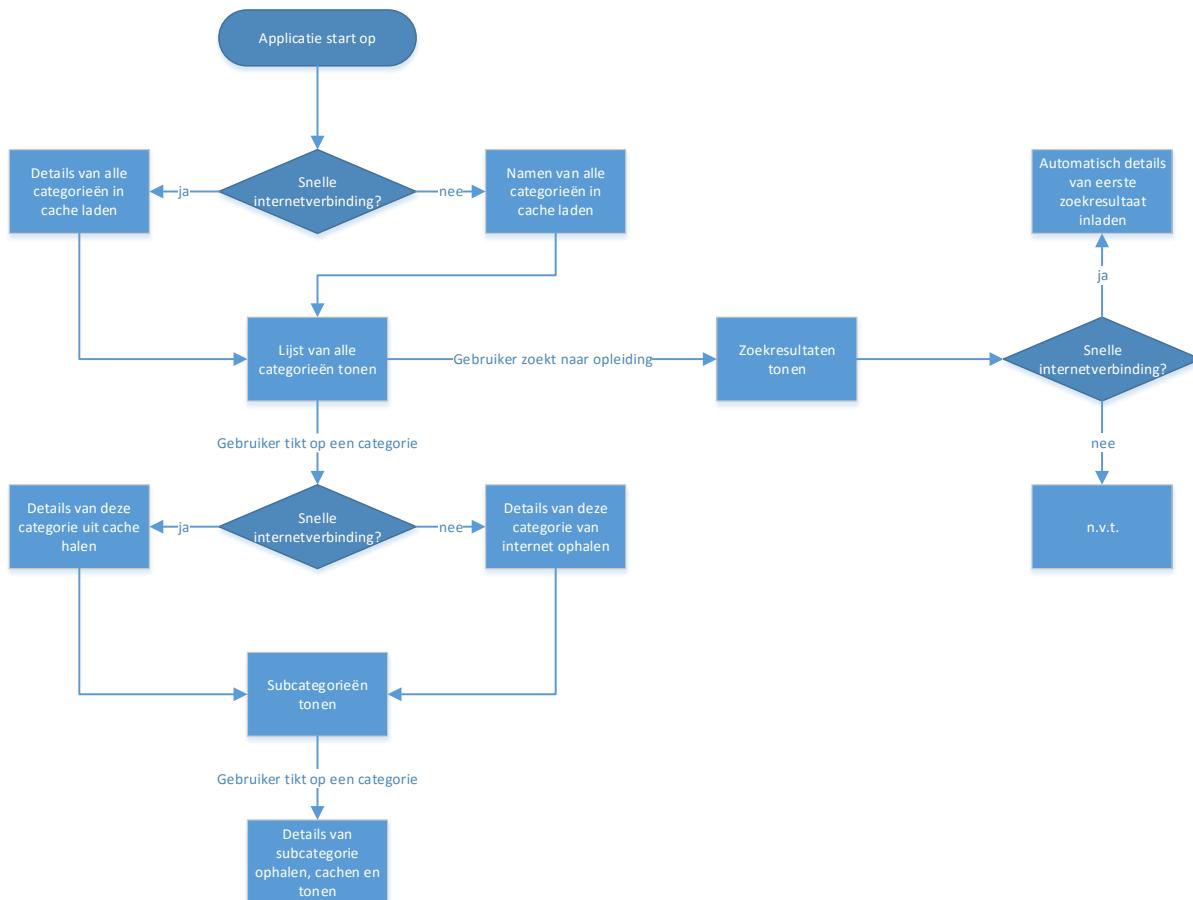
Op Windows Phone werd ook het design nog lichtjes gewijzigd. De statusbalk heeft de blauwe kleur van de RealDolmen huisstijl en het icoontje volgt de designpatronen van Windows Phone. Dit is niet mogelijk of toepasbaar op Android en iOS.

Sprint 3 is vlot van start gegaan. Er kwam echter wat werk bij door een bug op Android waarbij de view models niet correct gekoppeld worden aan de views (het visuele design van de Android app). Er is verder onderzoek nodig om deze bug te kunnen oplossen. Hetzelfde probleem deed zich ook voor bij de iOS app, maar is ondertussen verholpen.

## 9.2.6 Week 6

Deze week lag de nadruk opnieuw op features afwerken die voor sprint 3 moesten af zijn. Toen ik aan de functie om RealDolmen Education te contacteren kwam, merkte ik dat ik de sprintindeling lichtjes zou moeten aanpassen. Deze functie is namelijk gekoppeld aan een opleiding en kan dus pas geïmplementeerd worden als het scherm af is waarbij opleidingen weergegeven worden. Daarom werd deze feature uitgesteld naar sprint 4.

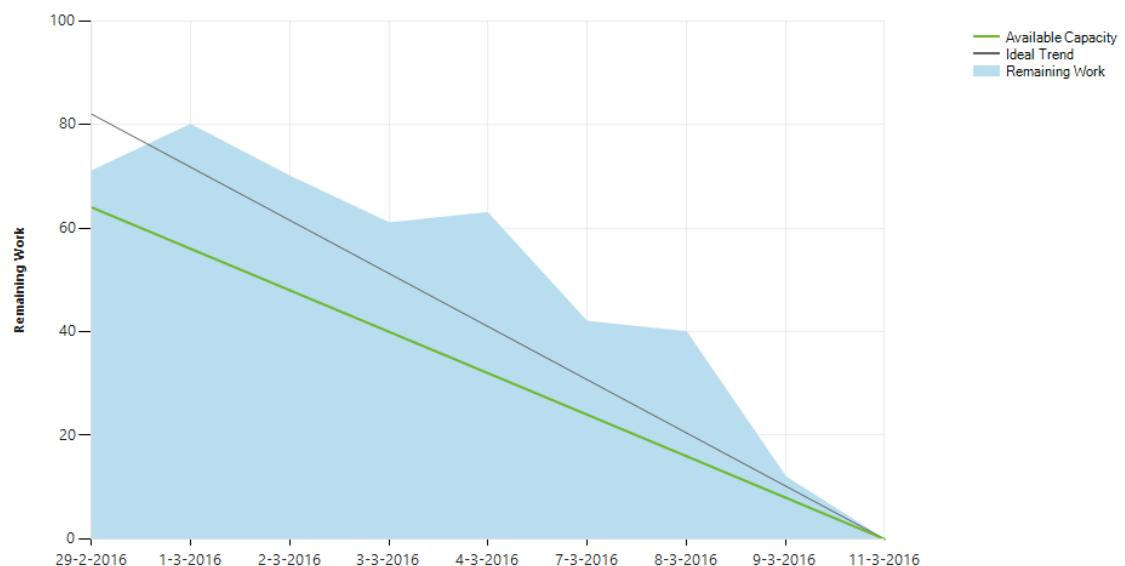
In de plaats werd er dan tijd besteed aan het opstellen van een betere service-laag. De volgende flowchart brengt deze in kaart:



Figuur 15: verduidelijking van de werking van caching

Om te bepalen of de gebruiker een snelle internetverbinding heeft, wordt nagekeken of er gebruik wordt gemaakt van een mobiel netwerk of niet. Deze laatste zijn vaak beperkt in bandbreedte en verbruik. Om het verbruik van de gebruiker te beperken, worden mogelijk overbodige gegevens dus enkel over Wi-Fi of ethernet in cache geladen.

Alle backlog items gepland voor sprint 3 waren aan het einde van sprint 3 volledig afgewerkt.



Figuur 16: burndown chart week 6

Donderdag was geen werkdag aangezien er dan een jobbeurs op Odisee was met verplichte aanwezigheid.

### 9.2.7 Week 7

De sprint review van sprint 3 verliep vlot aangezien alle geplande backlog items volledig afgewerkt waren. Er moet natuurlijk wel rekening gehouden worden met de tussentijdse herindeling die in het vorige verslag vermeld werd. De mogelijkheden om contact op te nemen werden uitgesteld om voorrang te geven aan caching en het weergeven van opleidingen.

Bij sprint 4 werd hier dan ook meteen weer opgepikt. Het grootste backlog item is het weergeven van gedetailleerde informatie van een opleiding. Tijdens de ontwikkeling hiervan werd een lacune in de API opgemerkt. De API is namelijk niet in staat om deze gegevens aan te bieden. Volgens de documentatie zou dit moeten werken, maar in de broncode van de API wordt nog een `//TODO` vermeld op de plaats waar deze functie zich zou moeten bevinden.

Dit betekent dat het in de app niet mogelijk is om de weergave van deze gegevens te testen. Tijdens de ontwikkeling worden tijdelijke testgegevens gebruikt, maar deze zijn niet beschikbaar tijdens het uitvoeren of demonstreren van de applicatie. Het inbouwen van deze functie in de API valt buiten de scope van het project waardoor dit niet meteen op de backlog komt. Mogelijks wordt hier wel nog naar gekeken bij de afwerking van sprint 6.

Het ontwerp van de schermen voor de opleidingsdetails is ook zonder dit obstakel een onderschat backlog item. Vooral op Android verliep dit proces langzamer dan verwacht omdat er heel veel verschillende details zijn die elk op een ordelijke manier weergegeven moeten worden. Ook de code hiervoor blijft best zo ordelijk mogelijk. Op Windows Phone lukt dit heel eenvoudig en op Android verloopt dit proces moeizaam. Vrijdag werd begonnen aan de implementatie hiervan op iOS. Hier zijn wat meer mogelijkheden om dit visueel duidelijker weer te geven, maar dit vereist ook wat onderzoek naar de werking van enkele minder vaak gebruikte visuele componenten.

De sprint had in principe iets beter gepland kunnen zijn. Het was de eerste keer dat er een dergelijke hoeveelheid aan designwerk aan te pas kwam wat waarschijnlijk de verklaring is voor deze inschattingsfout.

### 9.2.8 Week 8

Scrumgewijs was dit een slechte week, maar wat mijn ervaring en kennis betreft was afgelopen week misschien de meest leerrijke tot nu toe. De belangrijkste taak in deze sprint was de weergave van details van een opleiding (titel, beschrijving, prijs enz.). Deze taak heb ik op zich wel wat onderschat. Er kwam daarnaast ook wat refactoring werk bij kijken.

Als eerste ben ik begonnen met de lay-out van dit scherm op elk platform. De opbouw hiervan is platform-specifiek en kost het minste tijd bij Windows Phone en het meeste tijd bij Android. Vervolgens moest de lay-out verbonden worden met het view model waar alle commando's en gegevens over een opleiding in zit.

Windows Phone heeft hiervoor de ingebouwde bindingmogelijkheden, maar bij Android en iOS moet berroep gedaan worden op de binding helpers van MVVM Light. Bij iOS werkte dit zonder problemen, maar bij Android had ik er heel wat problemen mee. Ik kreeg regelmatig een TargetInvocationException, InvalidOperationException en anders lukte de binding wel, maar werd hij niet bijgewerkt wanneer nieuwe gegevens ingeladen waren. Om dit verder uit te doen, heb ik een blogpost over geschreven die beschikbaar is op <https://chipsncookies.com/2016/fix-common-binding-errors-with-mvvm-light-on-xamarin/>. Overigens had ik enkele weken geleden op dezelfde blog ook al een post geplaatst over dependency injection met Autofac: <https://chipsncookies.com/2016/dependency-injection-with-autofac-and-mvvm-light-in-xamarin/>

Het gevolg hiervan was dat er aanpassingen vereist waren aan alle views op Android, alle views op iOS en de baseclass van de views op Windows Phone. Ook de view models moesten aangepast worden. Het loste enkele openstaande bugs op en er kwamen geen exceptions meer naar boven, maar de opleidingspagina op Android werd nog steeds niet bijgewerkt. Dit is een bug die momenteel nog open staat en vermoedelijk enkel opgelost kan worden door ofwel geen binding te gebruiken, maar event handlers, ofwel wachten op een update van MVVM Light. Laurent Bugnion, de ontwikkelaar van MVVM Light, liet me weten dat dergelijke problemen waarschijnlijk wel opgelost gingen zijn in de volgende versie.

De meeste taken van sprint 4 werden overgedragen naar sprint 5. Dit brengt de uitwerking van de requirements niet in het gevaar, maar het zorgt ervoor dat er in sprint 6 waarschijnlijk geen tijd overblijft voor enkele nice-to-haves.

## 9.2.9 Week 9

De week begon met een vakantiedag en vanaf woensdag vond het evenement Microsoft Build plaats wat beschreven werd in een apart verslag.

Zoals verduidelijkt in het verslag van week 8 waren er nog heel wat openstaande taken uit sprint 4 die overgedragen moesten worden naar sprint 5. De voornaamste taken voor sprint 5 zijn het weergeven van de sessies van een opleiding en mogelijkheden om feedback te geven, contact op te nemen en zich in te schrijven.

De API die voor de sessies gebruikt dient te worden vertoonde nog enkele bugs en gaf ook nooit gedetailleerde informatie over een sessie terug. Dit bracht extra werk met zich mee; er moesten testgegevens gemaakt worden. Daarnaast is de API zo opgebouwd dat er een sessie vereist is om zich te kunnen inschrijven, contact opnemen, feedback geven... Dit is niet helemaal analoog aan de werking van de website.

Aan het einde van de week was de weergave van informatie van sessies afgewerkt op Windows Phone en gedeeltelijk op Android.

Naar aanleiding van Microsoft Build, heeft Xamarin een update uitgebracht die heel wat bugs opgelost heeft en de ontwikkeling met Xamarin.Android aanzienlijk vereenvoudigen. IntelliSense werkt nu volledig bij Android .axml-bestanden (lay-out).

Screenshots van de voorlopige versie van de app zijn terug te vinden in de gedeelde map op Google Drive.

### 9.2.10 Week 10

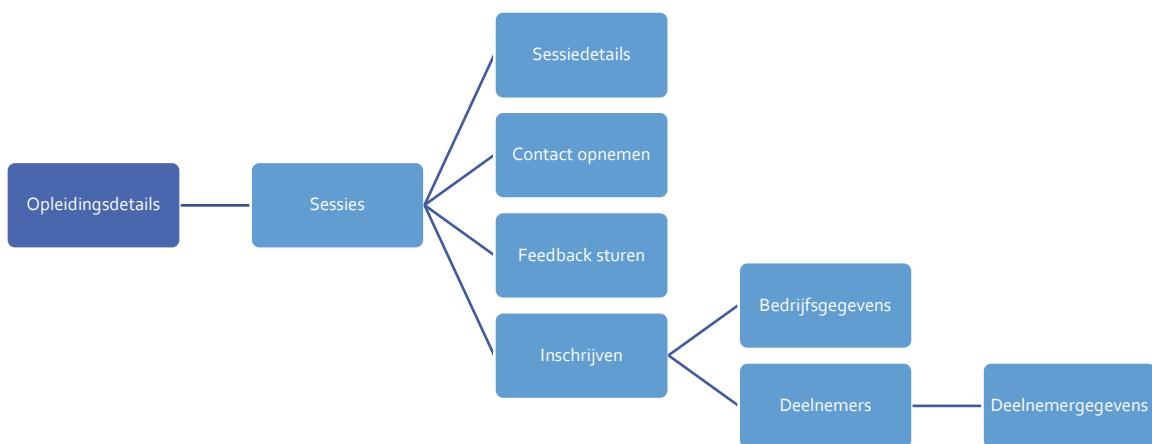
Tijdens de afgelopen week werden de views voor de sessies afgewerkt en vonden er verschillende evenementen plaats.

#### 9.2.10.1 Stageproject

Ondertussen ben ik aanbeland bij de onderste laag van het datamodel.



Figuur 17: reeds afgewerkte delen



Figuur 18: af te werken delen

Het moest eveneens mogelijk zijn om in te schrijven voor een sessie. Hier kwam meer werk bij kijken dan oorspronkelijk verwacht. Het visuele aspect is hier sowieso al vrij complex aangezien er veel details over de inschrijving gevraagd worden en er daarnaast ook nog een lijst van deelnemers opgesteld moet kunnen worden.

Op Windows Phone zijn deze schermen ondertussen volledig afgewerkt, waardoor ook meteen de business logica afgewerkt is. Op iOS zijn alle schermen afgewerkt, behalve het inschrijfscherm. Op Android zijn de sessiedetails afgewerkt, maar moeten de drie contactmogelijkheden nog aangemaakt worden.

#### 9.2.10.2 Evenementen

##### 9.2.10.2.1 Sollicitatie

Elke stagiair bij RealDolmen legt normaal gezien een sollicitatiegesprek af waarna mogelijks een jobaanbod volgt. Bij mij was dit moment donderdagnamiddag ingepland waarna effectief meteen een jobvoorstel volgde.

### 9.2.10.2.2 Xamarin meetup

Dinsdagavond hadden Microsoft en enkele Belgische Xamarin-ontwikkelaars de handen in elkaar geslagen voor een meetup voor Xamarin developers. Er waren drie interessante sessies georganiseerd waarover meer details in een apart verslag.

### 9.2.11 Week 11

De lay-out van de schermen om in te schrijven, feedback te geven en RealDolmen Education te contacteren, bleek, zoals vermeld in het vorige verslag, ingewikkelder dan eerst verwacht. Dit was dus de prioriteit van deze week.

Om dit probleem aan te pakken op iOS heb ik gebruik gemaakt van een tabel. Alle lay-outs die min of meer op een lijst lijken, kunnen als tabellen weergegeven worden. Een tabel bestaat hier namelijk altijd maar uit één kolom en de rijen mogen verschillende types cellen bevatten. De cellen bevatten dus afwisselend tekst (bv. "Voornaam: ") en een tekstveld waar de gebruiker iets kan invoeren.

Op Android was het probleem eenvoudiger op te lossen, gelijkaardig aan de XAML-versie van Windows Phone.

De rest van sprint 6 gaat verder over het afwerken van de applicatie en het meertalig maken van de applicatie. Omdat ik in vorige sprints hier altijd rekening mee gehouden heb, wordt dit waarschijnlijk een eenvoudig proces. Alle tekstvelden staan namelijk op centrale locaties opgeslagen die gemakkelijk kunnen omgezet worden naar de drie mobiele platformen.

Maandag was er ten slotte ook nog een evenement van RealDolmen waar alle .NET software engineers inclusief de stagairs op uitgenodigd waren. Om 15h spraken de managers het team toe en hadden ze het over de missie, visie en waarden van RealDolmen. Om 17h nam de RealDolmen Microsoft Community het woord over waarbij ze het hadden over alle activiteiten en dergelijke die zij organiseren voor hun .NET-ontwikkelaars. Ook ik mocht daar even het woord voeren en het kort hebben over mijn stage en ervaringen bij RealDolmen.

De slides die hierbij gebruikt werden, zullen geupload worden in de gedeelde Google Drive.

### 9.2.12 Week 12

Tijdens de laatste week van mijn stage kwamen vooral de vertalingen aan bod. Hiervoor heb ik gebruik gemaakt van de tool POEditor.com. Deze dienst vereenvoudigt het vertalen van cross-platform apps. Vertalingen werken namelijk anders op elk platform:

- Android: In de hoofdmap van het project moet een map 'resources' zitten met daarin een map 'values-CODE'. Code is hier de ISO 639-1 taalcode. In die map moet vervolgens een bestand genaamd 'strings.xml' zitten waarin de teksten zitten voor die taal. De hooftaal zit in een map 'values' zonder taalcode.
- iOS: In de hoofdmap van het project zit een map 'Base.lproj' met daarin een bestand 'Localizable.strings' die de teksten voor de hooftaal bevatten. Verder moet er een map 'CODE.lproj' bestaan met daarin opnieuw een bestand 'Localizable.strings' met de vertalingen in die taal. Code is ook hier de ISO 639-1 taalcode. Een storyboard vertalen kan door de translation identifiers te gebruiken die in een storyboard gevonden kunnen worden. Die zitten dan in 'Storyboard.strings' in een .lproj-map waarbij 'Storyboard' de naam is van het te vertalen storyboard.
- Windows Phone: In de hoofdmap van het project zit een map 'Strings' met daaronder een map die als naam een ISO 639-1-taalcode heeft. Daaronder zit telkens een bestand 'Resources.resw' met de vertalingen.
- Om ook in code overal dezelfde verwijzingen te kunnen gebruiken voor een tekstje uit de vertalingen, heb ik gebruik gemaakt van een enum met daarin alle mogelijke 'keys' voor vertalingen.

POEditor.com kan heel eenvoudig elk van deze formaten importeren en naar elk van deze formaten exporteren. Bovendien kan je online de vertalingen ingeven of alles automatisch door Google Translate laten vertalen. Soms kwam er nadien wat werk aan te pas met reguliere expressies om de vertalingen op elk platform goed te herkennen, maar meestal kon ik de vertalingen in 15 minuten op alle platformen bijwerken.

In het tweede deel van de week heb ik een presentatie voorbereid die ik op mijn laatste dag aan Nico Vermeir (stagebegeleider), Tom Knockaert (manager RealDolmen Education) en een medewerker van RealDolmen Education gegeven heb. Hier hoorde ook een demonstratie van de applicatie bij.

De feedback was over het algemeen wel goed. Er waren nog enkele bugs in de app die door tijdgebrek of door problemen met gebruikte componenten niet tijdig opgelost waren, maar die konden later nog opgelost worden. De aanwezigen van RealDolmen Education waren soms iets minder tevreden met de inhoud van de applicatie. De gegevens die de API publiekelijk beschikbaar stelt, mogen namelijk niet allemaal publiek zichtbaar zijn in de applicatie. Dit was te wijten aan een verkeerde interpretatie van het gegevensmodel van de API. Had deze feedback wat vroeger gekomen, dan had dit waarschijnlijk wel snel verholpen kunnen worden.

## 9.3 Evenementverslagen

### 9.3.1 Git and Beyond

23 februari 2016, 3 Oak Academy, Kontich

#### 9.3.1.1 Inleiding

Het evenement werd georganiseerd door MADN.

MADN staat voor Metro App Developer Network. Toen Microsoft Windows 8 lanceerde, lanceerden ze ook meteen een volledig nieuwe manier om apps voor Windows te maken. De klassieke Windows Forms en WPF-apps waren uit den boze en de nieuwe stijl van Windows apps heette Metro. De naam veranderde later naar Modern UI, maar dat hield enkele Belgische softwareontwikkelaars niet tegen om zich te verenigen in een nieuwe user group, gelijkaardig aan AZUG en VISUG.

Ze organiseren maandelijks evenementen waar sessies en dergelijke gegeven worden om ontwikkelaars te helpen betere apps te maken en de laatste nieuwe methodieken uit de doeken te doen.

In deze sessie geeft Jan De Dobbeleer uitleg over het gebruik van git. Git is een distributed version control system. Het wordt voornamelijk gebruikt om in teamverband aan de broncode van software te werken en om een team meer agile te maken.

#### 9.3.1.2 Agenda

- 18h: ontvangst
- 18h30: sessie over Git door Jan De Dobbeleer
- 21h: networking & drinks

#### 9.3.1.3 Verslag

##### 9.3.1.3.1 Verworven kennis

Zelf was ik al vrij goed bekend met git, maar de kennis van Jan omtrent git gaat veel dieper. De sessie begon met een algemene inleiding tot de principes van git.

Git werkt namelijk veel efficiënter wat opslag en branching betreft. Het staat je toe om verschillende versies van je code te hebben die je nadien kan samenvoegen. Zo kan je een versie hebben die overeenkomt met de code die in productie gebruikt wordt en eentje waarop ontwikkeld wordt. Een bugfix kan dan heel snel op de productieversie gebeuren terwijl een nieuwe feature enkel aan de ontwikkelingsversie toegevoegd wordt.

Naar het einde toe, liet Jan wat geavanceerdere mogelijkheden van Git zien. Zo kan je snel ontdekken welk stukje code mogelijke problemen veroorzaakt heeft.

##### 9.3.1.3.2 Persoonlijke mening en ervaringen

Aangezien ik git gebruik tijdens zo goed als elk project, inclusief mijn stage, was dit een zeer leerrijke sessie. Vooraf dacht ik dat ik weinig nieuws ging leren, maar deze sessie heeft me toch opnieuw van mijn keuze voor git overtuigd en heel wat nieuws bijgebracht over de geavanceerde mogelijkheden van git.

Terwijl ik vroeger een GUI gebruikte om met git te werken, ben ik na deze sessie overgestapt op een command line interface. Het is een veel krachtigere tool en je kan er veel sneller mee bijleren.

Zoals steeds dus een interessante sessie van MADN.

### 9.3.1.3.3 TI-competenties

De volgende competenties en kerndoelen kwamen aan bod:

- 7. Op een adequate wijze IT-oplossingen configureren, beveiligen en aanpassen zodat deze beantwoorden aan de veranderende behoeften van de organisatie.
  - 7.3 Toepassingen, databanken en systemen als gevolg van wijzigende omstandigheden aanpassen.
- 9. IT-taken kwaliteitsvol uitvoeren zodat het resultaat voldoet aan de eisen van een steeds wisselende omgeving.
  - 9.1 Zelfstandig nieuwe IT-concepten en principes leren hanteren.
  - 9.5 Zelfstandig een kwaliteitszorgsysteem toepassen dat de continuïteit van de IT-processen garandeert.

### 9.3.2 Microsoft Build 2016

30 maart 2016 – 01 april 2016, Ordina, Mechelen

#### 9.3.2.1 Inleiding

Build is de jaarlijkse conferentie van Microsoft waarin het bedrijf laat zien wat ze het afgelopen jaar verworven hebben en gepland hebben voor het komende jaar. Het is een conferentie voor softwareontwikkelaars die werken met Microsoftproducten. Er wordt gedemonstreerd hoe een bepaalde technologie zijn nut kan bewijzen en hoe softwareontwikkelaars hiermee aan de slag kunnen.

VISUG (Visual Studio User Group) is een groep Belgische softwareontwikkelaars die sessies organiseren rond software die je met Microsoft Visual Studio kan ontwikkelen.

MADN staat voor Metro App Developer Network. Toen Microsoft Windows 8 lanceerde, lanceerden ze ook meteen een volledig nieuwe manier om apps voor Windows te maken. De klassieke Windows Forms en WPF-apps waren uit den boze en de nieuwe stijl van Windows apps heette Metro. De naam veranderde later naar Modern UI, maar dat hield enkele Belgische softwareontwikkelaars niet tegen om zich te verenigen in een nieuwe user group.

Ze worden gesponsord door grote IT-bedrijven actief op de Belgische markt die zich focussen op softwareontwikkeling met het .NET platform: Microsoft, RealDolmen, Ordina, Cegeka, Cronos, JetBrains, Capgemini...

VISUG en MADN organiseerden op 30 maart een evenement waarbij de keynote van Microsoft Build op groot scherm geprojecteerd wordt. Er waren broodjes, pizza en frisdrank voorzien en er was eveneens tijd voor een leuke babbel achteraf.

Microsoft Build ging nog enkele dagen door en er waren de hele dag door (Amerikaanse tijd) sessies georganiseerd.

#### 9.3.2.2 Agenda & sprekers

Zie <http://channel9.msdn.com>

#### 9.3.2.3 Verslag

##### 9.3.2.3.1 Verworven kennis

De onderwerpen die aan bod kwamen waren heel uiteenlopend. Het evenement begon met de nieuwe mogelijkheden van Windows 10 die eraan zitten te komen, ging over naar Xamarin, hologrammen (augmented reality) met Microsoft HoloLens, artificiële intelligentie met bots, spraak- en beeldherkenning...

In verband met mijn stage was natuurlijk vooral het Xamaringedeelte interessant. Microsoft heeft 10 dagen voor het evenement officieel Xamarin overgekocht. Tijdens de keynote van dag 2 kondigde Microsoft dan ook aan wat de plannen hieromtrent zijn. Xamarin kostte vroeger \$1000 per jaar per ontwikkelaar per platform (Android/iOS). Tijdens het evenement kondigden ze dan ook aan dat Xamarin gratis bij alle edities van Microsoft Visual Studio komt te zitten, ook bij de gratis Community edition. Daarnaast wordt alle broncode van Xamarin gepubliceerd op GitHub onder de MIT-licentie (vrij van auteursrecht).

##### 9.3.2.3.2 Persoonlijke mening en ervaringen

Het Xamarin-nieuws is geweldig voor het hele .NET-platform aangezien dit talloze nieuwe ontwikkelaars en klanten zal aantrekken. De toekomst van Xamarin is nu verder verzekerd.

Het evenement van VISUG was heel aangenaam en ik heb interessante discussies kunnen voeren met andere ontwikkelaars over de toekomst van het .NET-platform. Ook uit de livestreams van Microsoft Build heb ik talloze zaken bijgeleerd.

### 9.3.2.3.3 TI-competenties

De volgende competenties en kerndoelen kwamen aan bod:

- 7. Op een adequate wijze IT-oplossingen configureren, beveiligen en aanpassen zodat deze beantwoorden aan de veranderende behoeften van de organisatie.
  - 7.3 Toepassingen, databanken en systemen als gevolg van wijzigende omstandigheden aanpassen.
- 9. IT-taken kwaliteitsvol uitvoeren zodat het resultaat voldoet aan de eisen van een steeds wisselende omgeving.
  - 9.1 Zelfstandig nieuwe IT-concepten en principes leren hanteren.

### 9.3.3 Xamarin meetup

*5 april 2016, Microsoft, Zaventem*

#### 9.3.3.1 Inleiding

Na de aankondiging dat Microsoft Xamarin overneemt en volledig gratis en vrij van auteursrecht maakt, leek een update over de stand van zaken rond de integratie van Xamarin wel aangewezen. Dat is dan ook wat Microsoft tijdens de jaarlijkse Build-conferentie gedaan heeft. Ook de Belgische gemeenschap van Xamarin-ontwikkelaars moest niet onderdoen, want Microsoft België organiseerde eveneens een meetup.

De meetup werd georganiseerd door enkele Xamarin-ontwikkelaars, enkele mensen van Xamarin zelf en Microsoft België en ging door in de zalen van Microsoft België in Zaventem. Het bestond uit drie sessies die elk dieper in gingen op Xamarin of een verwante technologie.

#### 9.3.3.2 Agenda

- 18h30: ontvangst
- 19h: inleiding door Jan Tielens (Microsoft), Jan Van De Poel & Steve Gosserie
- 19h30: Reactive eXtensions (ReactiveX / Rx) – Benoit Jadinot (RiseUp Mobile)
- 20h: de toekomst van Xamarin – Mike James (Xamarin / Microsoft)
- 20h45: pauze
- 21h: i18n voor Xamarin.Forms met Vernacular – Stéphane Delcroix (Mi8 / Xamarin)
- 22h: networking drink
- 22h30: einde

#### 9.3.3.3 Verslag

##### 9.3.3.3.1 Verworven kennis

De drie sessies waren enorm leerrijk. Rx is een framework dat bij Microsoft ontworpen werd door de hoofdontwikkelaar van LINQ. De mogelijkheden van Rx zijn eindeloos, maar volgende tabel legt goed uit wat het doel ervan is.

	Enkelvoudige operaties	Werken met collecties
Synchroon	Standaardmethodes...	LINQ
Asynchroon	Async/await	ReactiveX

Figuur 19: verhouding van Rx tot andere componenten in .NET

Alles wat met LINQ op een (min of meer) synchrone manier mogelijk is, kan met ReactiveX op een volledige asynchrone manier. Daarnaast is het ook ideaal om sommige events af te handelen of gegevens op te halen en te verwerken. De toepassingen liggen vooral bij applicaties die altijd responsief moeten blijven (mobiele apps, desktop apps), maar het kan in principe over het hele .NET platform gebruikt worden (ASP.NET, console apps, forms...). Rx maakt gebruik van het observer-observable-patroon (verder uitgelegd in 9.7.3 OBSERVABLE-OBSERVER).

Mike James gaf aan de hand van enkele voorbeelden aan hoe Xamarin verwacht dat apps geschreven moeten worden. Zowel de architectuur (Model-View-ViewModel) als de manier van werken (Visual Studio / Xamarin Studio) werd uit de doeken gedaan.

Tijdens de laatste sessie werd aangetoond hoe Xamarin.Forms beter gebruikt kan worden. Het ging zowel over performantie als over de vertalingen van applicaties.

#### 9.3.3.2 Persoonlijke mening en ervaringen

De sessie over Rx was heel interessant, maar het ging eerder over een korte introductie waarna je zelf aan de slag kan om ermee te leren werken. De aanpak verschilt sterk van wat men gewoon is en er is nog veel studiewerk vereist vooraleer een ontwikkelaar er echt mee aan de slag kan.

De laatste sessie was redelijk interessant, maar ging de mist in door de vermoeide spreker die het Engels nauwelijks bekwaam was.

De meest leerrijke sessie was die van Mike James. Achteraf ben ik ook nog even met hem gaan praten en zijn aanpak verschilt van wat je vaak terugvindt als "beste aanpak" op het web. Zo gebruikt hij geen frameworks voor MVVM, maar zet hij de volledige architectuur zelf op. Dit maakt zijn code eenvoudiger en makkelijker aan te passen.

Kortom, voor herhaling vatbaar!

#### 9.3.3.3 TI-competenties

De volgende competenties en kerndoelen kwamen aan bod:

- 7. Op een adequate wijze IT-oplossingen configureren, beveiligen en aanpassen zodat deze beantwoorden aan de veranderende behoeften van de organisatie.
  - 7.3 Toepassingen, databanken en systemen als gevolg van wijzigende omstandigheden aanpassen.
- 9. IT-taken kwaliteitsvol uitvoeren zodat het resultaat voldoet aan de eisen van een steeds wisselende omgeving.
  - 9.1 Zelfstandig nieuwe IT-concepten en principes leren hanteren.

### 9.3.4 Global Azure Bootcamp 2016

16 april 2016, RealDolmen, Huizingen

#### 9.3.4.1 Inleiding

De Global Azure Bootcamp is een evenement waarbij Azure user groups en Microsoftpartners wereldwijd op hetzelfde moment enkele sessies organiseren. Ook de Belgische user group, AZUG, nam hieraan deel en organiseerde een evenement bij RealDolmen met 5 interessante sessies, catering en enkele leuke prijzen.

#### 9.3.4.2 Agenda & sprekers

- 8h30: ontvangst / ontbijt
- 9h15: inleiding (Wesley Cabus)
- 9h30: Not Steve, Just Jobs: Azure Web Jobs (Kris Van Der Mast)
- 10h20: Docker Orchestration on Azure with Rancher (Karim Vaes)
- 11h20: IoT Hub with UWP apps (Nico Vermeir)
- 12h35: lunch break
- 13h30: Hybrid integration with Azure (Glenn Colpaert)
- 14h45: Polyglot Persistence with Azure (Charalampos Karypidis)
- 16h: Using Azure to simplify your disaster recovery strategy (Nichola Van De Voorde & Wouter Gevaert)
- 16h50: dankwoord
- 17h15: einde

#### 9.3.4.3 Verslag

##### 9.3.4.3.1 Verworven kennis

Tijdens de eerste sessie werd een voorbeeld gegeven van hoe periodieke taken die gekoppeld zijn aan een web app, eenvoudig ingepland en uitgevoerd kunnen worden met Azure Web Jobs.

Nadien kwam Docker aan bod. Dit is een platform voor apps die in containers draaien en de opvolger van klassieke virtualisatie in virtuele machines. Het zorgt ervoor dat een systeem zoals een web app altijd in dezelfde configuratie kan werken en dat deze configuratie volledig aanpasbaar is via één script. Daardoor is het ook mogelijk om op enkele seconden tientallen extra instanties van de applicatie op te starten die dan naast elkaar kunnen draaien om de belasting wat te verdelen.

In de derde sessie deed mijn stagebegeleider uit de doeken hoe the internet of things kan gekoppeld worden aan Azure en hoe zo (mobiele) applicaties gebruikt kunnen worden om deze te bedienen.

Na de middagpauze gingen we verder met een sessie waarin gedemonstreerd werd hoe Azure kan geïntegreerd worden met heel wat bestaande systemen en applicaties. Het ging vooral om webomgevingen die door bedrijven gebruikt worden zoals CRM- en ERP-systemen.

Vervolgens kwam een Griekse spreker vertellen wat er vandaag allemaal mogelijk is met Azure om gegevens op te slaan. Klassieke RDBMS'en worden alsmaar vaker vervangen door NoSQL-databases.

Tenslotte deden twee systeembeheerders hun verhaal over hoe ze Azure gebruikt hebben als disaster recovery systeem voor als plots heel wat servers in een datacenter het laten afweten. Azure kan dan op korte tijd enkele virtuele machines of containers opstarten die het werk tijdelijk overnemen.

#### 9.3.4.3.2 Persoonlijke mening en ervaringen

De verschillende sessies waren allemaal gerelateerd aan het Microsoft Azure-platform, maar toch heel uiteenlopend wat onderwerp betreft. Behalve de laatste sessie, waren ze allemaal gericht op softwareontwikkelaars die het .NET-platform gebruiken. Dit toonde dan ook aan hoe groot de cloud van Microsoft geworden is, net als alle mogelijkheden. Azure heeft het grootste aantal datacenters ter wereld en groeit nog steeds verder.

Er was geen rechtstreekse link met de stage, behalve dan dat elke mobiele app gebruik maakt van de cloud voor de opslag en verwerking van gegevens.

#### 9.3.4.3.3 TI-competenties

De volgende competenties en kerndoelen kwamen aan bod:

- 7. Op een adequate wijze IT-oplossingen configureren, beveiligen en aanpassen zodat deze beantwoorden aan de veranderende behoeften van de organisatie.
  - 7.1 Systemen, toepassingen of databanken configureren in wijzigende omstandigheden.
  - 7.3 Toepassingen, databanken en systemen als gevolg van wijzigende omstandigheden aanpassen.
- 9. IT-taken kwaliteitsvol uitvoeren zodat het resultaat voldoet aan de eisen van een steeds wisselende omgeving.
  - 9.1 Zelfstandig nieuwe IT-concepten en principes leren hanteren.
  - 9.2 Zelfstandig IT-taken uitvoeren op een efficiënte en doeltreffende manier.
  - 9.3 Een gebruiksvriendelijke IT-oplossing uitwerken die aanpasbaar is aan de veranderende behoeften.

### 9.3.5 Xamarin Evolve

27 april 2016, Hyatt Regency, Orlando, FL, VS

#### 9.3.5.1 Inleiding

Xamarin Evolve is dé jaarlijkse conferentie voor ontwikkelaars en partners die met Xamarin werken. Zoals elke andere ontwikkelaarsconferentie begint het met een keynote door onder andere de CEO en volgen er nadien heel wat sessies die dieper ingaan op nieuwe mogelijkheden van het Xamarin-platform.

Evolve 2016 gaat door in Orlando, Florida, maar is live te volgen via een videostream.

#### 9.3.5.2 Agenda

- 15h-16h30: keynote
  - Nat Friedman, CEO & Co-Founder, Xamarin Inc.
  - Miguel De Icaza, CTO & Co-Founder, Xamarin Inc.
  - Nish Anil, Developer Evangelist, Xamarin Inc.
  - James Montemagno, Developer Evangelist, Xamarin Inc.
  - Nina Vyedin, Product Manager, Xamarin Inc.
  - Donovan Brown, Senior DevOps Program Manager, Microsoft Corp.
- Sessies (agenda op <https://evolve.xamarin.com/live> )

#### 9.3.5.3 Verslag

##### 9.3.5.3.1 Verworven kennis

Xamarin is sinds hun overname door Microsoft enorm gegroeid. De interesse in het platform is op enkele maanden tijd verdrievoudigd. Tijdens Microsoft Build werden al enkele nieuwe mogelijkheden aangekondigd, maar ook tijdens de keynote kregen we nog een voorproefje van wat volgende versies van het product te bieden hebben.

Zo plannen ze een iOS-simulator voor Visual Studio, een previewer voor Xamarin.Forms en USB-verbindingen van Visual Studio naar een MacBook voor Xamarin.iOS. Verder zal de volgende versie van Xamarin Studio ondersteuning bieden voor Roslyn en C# 6, een donker design, 64-bit processoren en F#. Daarnaast wordt Xamarin volledig open source. De broncode van het volledige platform is te vinden op GitHub.

Ook rond DevOps zitten er veel nieuwe functies in de pijplijn. Apps testen wordt eenvoudiger via Xamarin Test Cloud en HockeyApp. Je zal zelfs je app live op elke mogelijke smartphone kunnen testen via Visual Studio.

##### 9.3.5.3.2 Persoonlijke mening en ervaringen

De toekomst lijkt alsmaar rooskleuriger te worden voor Xamarin. Evolve biedt een mooie context aan mijn stage en gaf wat nieuwe inzichten in hoe Xamarin gebruikt kan worden om knappere apps te ontwikkelen.

Wat ik persoonlijk het knapst vond was Xamarin Test Cloud Live. Xamarin geeft je volledige controle over een smartphone naar keuze uit de Xamarin Test Cloud en verbindt de debugger van Visual Studio met dit toestel. Je kan dus live debuggen op elk type smartphone, inclusief werkende breakpoints en IntelliTrace vanop honderden kilometers afstand. Deze nieuwe tools vallen onder de noemer 'DevOps'.

### 9.3.5.3.3 TI-competenties

De volgende competenties en kerndoelen kwamen aan bod:

- 9. IT-taken kwaliteitsvol uitvoeren zodat het resultaat voldoet aan de eisen van een steeds wisselende omgeving.
  - 9.1 Zelfstandig nieuwe IT-concepten en principes leren hanteren.
  - 9.3 Een gebruiksvriendelijke IT-oplossing uitwerken die aanpasbaar is aan de veranderende behoeften.

## 9.4 POP / competenties

### 9.4.1 Begin van de stage

#### 9.4.1.1 *Algemene competenties*

*Vlot functioneren in een professionele (internationale) omgeving*

Als softwareontwikkelaar werk je met heel wat mensen samen om een goed product af te leveren. Ook bij het project voor RealDolmen Education is dit het geval. Enerzijds zijn er dienstverlenende partners zoals de HR-afdeling en de interne helpdesk die respectievelijk een administratieve en een technische bijdrage moeten leveren zodat alles ter beschikking staat om het project te kunnen realiseren. Anderzijds werk je samen met de klant om samen naar de oplossing te werken waar alle partijen zich het meeste in kunnen vinden. Tenslotte kunnen collega's ook een belangrijke rol spelen en kan hun expertise tot nieuwe inzichten leiden.

Hier heb ik aan het begin van mijn stage weinig tot geen ervaring mee. Daarom geef ik mezelf een 5/10 op deze competentie.

*Relaties met collega's, opdrachtgevers en/of klanten onderhouden*

Het is belangrijk om regelmatig updates te geven over de voortgang van het project aan de opdrachtgever, vaak ook de 'product owner' genoemd. In dit geval is dit Tom Knockaert, manager van RealDolmen Education. Hij zal indien mogelijk tussentijdse feedback geven over de uitwerking van de mobiele app. Negatieve feedback kan dan waarschijnlijk bijkomend werk inhouden. Ook de stagebegeleiders evalueren tussentijds de voortgang van het project.

Daar ik tijdens mijn opleiding ook vele tussentijdse evaluaties gehad heb en projecten dan lichtjes heb moeten bijsturen, heb ik hier al enige ervaring mee. Hier geef ik mezelf een 6/10 op.

#### 9.4.1.2 *IT-competenties*

*Gegevens verzamelen, opslaan en ter beschikking stellen zodat deze op een correcte en gebruiksvriendelijke manier kunnen worden opgevraagd*

Dit omschrijft eigenlijk het doel van de applicatie. Gegevens van RealDolmen Education moeten op een gebruiksvriendelijke manier voor smartphones en tablets aangeboden worden. De huidige website doet dit niet optimaal en hier moet de mobiele applicatie een oplossing aan bieden.

Hier heb ik al heel wat ervaring mee. Als jobstudent heb ik al gelijkaardige applicaties ontwikkeld en dit kwam eveneens meerdere keren tijdens eerdere fasen van de opleiding aan bod. Daarom scoor ik mezelf hier 8/10.

*Nieuwe IT-oplossingen (autonom) uitwerken in overeenstemming met de verwachtingen van de opdrachtgever*

Bij dit project spelen verschillende partijen een rol. Zo is RealDolmen MS Community de rechtstreekse opdrachtgever, maar RealDolmen Education de eigenlijke klant. Ze hebben elk eigen verwachtingen en beoordelen andere aspecten. Met beide partijen moet dan ook op een andere manier rekening gehouden worden. Het was belangrijk om de opdracht goed te omschrijven, wat in de bachelorproef gebeurd is.

Tot aan de stage heb ik zelden voor concrete opdrachtgevers gewerkt met dergelijke specifieke eisen. Het lijkt me daarom een uitdaging en ik geef mezelf hier voorlopig een score van 7/10 op.

*Een IT-opdracht projectmatig en teamgericht aanpakken met respect voor de planning*

Er wordt gewerkt met scrum. De stage wordt gepland in 6 sprints van 2 weken. Aan de hand van Visual Studio Online<sup>6</sup> wordt de voortgang tussentijds bijgehouden. Dit stelt alle partijen in staat om de resterende hoeveelheid werk op te vragen en een stabiele vooruitgang te bewaken.

Zelf ben ik bij elk project heel georganiseerd te werk gegaan. De omvang van dit project is echter heel wat groter dan het gemiddelde project dat ik al afgewerkt heb. Daarom scoor ik mezelf hierop een 7/10.

#### 9.4.2 Einde van de stage

##### 9.4.2.1 *Algemene competenties*

###### *Vlot functioneren in een professionele (internationale) omgeving*

Tijdens mijn stage is dit enkele keren aan bod gekomen. Er was dagelijks contact met verschillende collega's en ik heb enkele keren beroep op hen gedaan. Dit was echter niet in dergelijke mate dat ik hier enorm veel nieuwe ervaring rond heb opgedaan. Daarom scoor ik mezelf nu 7/10 op deze competentie.

###### *Relaties met collega's, opdrachtgevers en/of klanten onderhouden*

Elke week had ik wel een sprintreview met Nico Vermeir (stagementor RealDolmen) en tijdens de stage heb ik twee keer afgesproken met Serge Van Cleynenbreugel (stagementor Odisee). Tom Knockaert van Real-Dolmen Education, de eigenlijke klant, werd tweewekelijks op de sprint review uitgenodigd. Hij kon spijtig genoeg zelden aanwezig zijn wegens andere verplichtingen. Omdat de communicatie met de opdrachtgever wel goed verliep, zou ik mezelf nu 8/10 geven op deze competentie.

##### 9.4.2.2 *IT-competenties*

###### *Gegevens verzamelen, opslaan en ter beschikking stellen zodat deze op een correcte en gebruiksvriendelijke manier kunnen worden opgevraagd*

De applicatie die tijdens de stage uitgewerkt werd, geeft de gegevens van op de website veel duidelijker weer aan de gebruiker dan wat ervoor het geval was met een smartphone en de bestaande website. Vanzelfsprekend is er altijd wel iets dat hier nog verder aan te verbeteren valt en daarom scoor ik mezelf hier nu 9/10 op.

###### *Nieuwe IT-oplossingen (autonom) uitwerken in overeenstemming met de verwachtingen van de opdrachtgever*

Het ging bij dit project om een volledig nieuwe oplossing. Er bestond al een website met dezelfde gegevens, maar er was nog geen mobiele toepassing. De uitwerking ervan gebeurde zelfstandig, met de nodige begeleiding van stagementors en zelden ook van collega's. Daarnaast werden alle vereisten uit het PID opgevolgd. Hiervoor maakten we gebruik van Visual Studio Team Services. De backlog werd hier ingevoerd zodat iedereen de voortgang van het project kon opvolgen.

Omdat ik deze competentie als geslaagd beschouw, maar er hier en daar nog wat kleine verbeteringen aan de applicatie mogelijk zijn, scoor ik mezelf 9/10 op deze competentie.

###### *Een IT-opdracht projectmatig en teamgericht aanpakken met respect voor de planning*

Net zoals bij de vorige competentie werd Visual Studio Team Services gebruikt om met de scrum-methode te werken. Dit is een methode om heel projectmatig te werk te gaan, zodat aan het einde van het project de

<sup>6</sup> [https://rdmscc.visualstudio.com/DefaultCollection/Stageproject%20Education%20Samuel/Stageproject%20Education%20Samuel%20Team/\\_dashboards](https://rdmscc.visualstudio.com/DefaultCollection/Stageproject%20Education%20Samuel/Stageproject%20Education%20Samuel%20Team/_dashboards)

meeste features / backlog items afgewerkt zijn. Indien dit niet gevuld zou worden, bestaat de kans dat aan het einde van het project geen enkele feature afgewerkt is, maar er wel aan elke feature gewerkt is geweest.

Daar ik dit strikt gevuld heb, maar ik in het begin wel nog moeilijkheden had met het inschatten van de hoeveelheid werk, scoor ik mezelf hier ook 9/10 op.

#### 9.4.3 Overzicht scores

Competentie	Score begin	Score einde
Vlot functioneren in een professionele (internationale) omgeving	5/10	7/10
Relaties met collega's, opdrachtgevers en/of klanten onderhouden	6/10	8/10
Gegevens verzamelen, opslaan en ter beschikking stellen zodat deze op een correcte en gebruiksvriendelijke manier kunnen worden opgevraagd	8/10	9/10
Nieuwe IT-oplossingen (autonom) uitwerken in overeenstemming met de verwachtingen van de opdrachtgever	7/10	9/10
Een IT-opdracht projectmatig en teamgericht aanpakken met respect voor de planning	7/10	9/10

Tabel 10: overzicht van de scores voor de competenties in het POP

#### 9.4.4 Technische concepten

Voorafgaand aan de stage had ik al ervaring met de volgende relevante technische concepten:

- De ontwikkeling van Android apps (zonder Material design)
- De ontwikkeling van Windows Store (Windows 8.1) apps
- Configuratie en gebruik van continue integratie
- Model-View-Controller (MVC) ontwerppatroon
- Unit testing
- Inversion of Control / Dependency Injection

Deze kennis heeft zeker geholpen tijdens de stage en tijdens het aanleren van nieuwe IT-concepten die tijdens de stage aan bod kwamen. Tijdens de stage heb ik de volgende nieuwe technische concepten bijgeleerd:

- De ontwikkeling van Android apps met Material design
- De ontwikkeling van Windows Phone 8.1 apps
- De ontwikkeling van iOS apps
- Configuratie en gebruik van continue integratie met Visual Studio Team Services en Apple OS X
- Ontwerppatronen: Model-View-ViewModel (MVVM), Command, Factory, Observable-Observer
- OAuth2 implementeren voor authenticatie
- Cross-platform apps met Xamarin

#### 9.4.5 Conclusie

Tijdens de stage heb ik op verschillende vlakken nieuwe kennis en ervaring opgedaan: projectmatig, technisch, communicatief, professionele samenwerking... De stage was dus zeer waardevol voor mijn persoonlijke ontwikkeling en er zijn meerdere competenties uit de opleiding Toegepaste Informatica aangewend.

Het was dan ook de bedoeling om de applicatie voor RealDolmen Education te ontwikkelen met moderne, meer geavanceerde principes en methodes en een ordelijke en volledig uitgewerkte softwarearchitectuur. De voornaamste uitdaging lag erin om zoveel mogelijk code te delen over de drie mobiele platformen heen.

Daarnaast was het belangrijk om op een projectmatige manier te werk te gaan. Tijdens de opleiding werden de mogelijke manieren uit de doeken gedaan. Zo is het bij beginnende softwareontwikkelaars vaak een uitdaging om de recente scrum-methode goed uit te voeren en niet te laten vervallen in de klassieke watervalaanpak.

Ten slotte hebben de tweewekelijkse sprint reviews en het dagelijkse contact met collega's bijgedragen aan de verwerving van een professionele manier van samenwerken.

## 9.5 Termenlijst

Onderstaande lijst geeft een overzicht en verklaringen van de technische termen die regelmatig in dit document voorkomen.

- **API (Application Programming Interface):** een API is aanspreekpunt voor softwareontwikkelaars dat gegevens of functies kan aanbieden. Dit kan gaan om het ophalen of verwerken van gegevens of het gebruik van functies uit andere software zoals een besturingssysteem. Een API wordt vaak aangeboden via het internet (dit gaat dan om een web API) of via een Software Development Kit (SDK).
- **Caching:** dit is een principe dat vaak wordt toegepast bij software die gegevens ophaalt of verwerkt van een externe bron zoals het internet. Vaak moeten de eindgebruikers van de software wachten totdat de gegevens zijn ingeladen. Om deze wachttijd te beperken of zelfs volledig uit te sluiten, worden de gegevens tijdelijk lokaal opgeslagen. Dan kan de software de gegevens lokaal ophalen wat veel sneller verloopt. In hoofdstuk 9.8 CACHING wordt dit verder toegelicht.
- **DI (Dependency Injection):** DI is een manier om Inversion of Control toe te passen. Hoe dit in zijn werk gaat, is uitgelegd bij 9.7.2 DEPENDENCY INVERSION.
- **HttpClient:** bij applicaties die gebouwd zijn met Microsofts .NET-platform kan een HttpClient gebruikt worden om gegevens over het internet te verzenden. Dit wordt vaak gebruikt voor communicatie met een web API.
- **IoC (Inversion of Control):** om ervoor te zorgen dat software later eenvoudig aangepast kan worden, wordt het aangeraden om geen vaste referenties bij te houden naar afhankelijkheden. Dit ontwerppatroon wordt verder verduidelijkt in 9.7.2 DEPENDENCY INVERSION.
- **JSON (JavaScript Object Notation):** om gegevens te versturen over internet, worden ze in deze notatie of in XML genoteerd. De conversie van objecten naar JSON en terug wordt automatisch uitgevoerd door een pakket als Json.NET (9.15.2.11 JSON.NET). JSON heeft het voordeel van heel weinig extra karakters of opmaak te vereisen om gegevens weer te geven.
- **MVVM (Model-View-ViewModel):** dit is een ontwerppatroon om code te structureren bij mobiele apps. Dit wordt verder uitgelegd in bijlage 9.7.1 MODEL-VIEW-VIEWMODEL.
- **NuGet package:** dit is een codebibliotheek die via het populaire NuGet-platform gratis beschikbaar gemaakt wordt aan anderen. NuGet is geïntegreerd in de ontwikkelomgeving Microsoft Visual Studio waardoor het heel eenvoudig is om dergelijke code te integreren in eigen software. De NuGet packages die gebruikt werden, zijn vermeld in bijlage 9.15.2 NUGET PACKAGES.
- **PCL (Portable Class Library):** bij de ontwikkeling van cross-platform applicaties wordt er geprobeerd om zoveel mogelijk code te delen over de platformen. Code die op meerdere platformen moet werken, moet in een PCL geplaatst worden. Een PCL heeft de eigenschap om code te kunnen compileren naar verschillende platformen. Er bestaan eveneens nog alternatieven voor een PCL, maar deze kwamen bij dit project niet aan bod.
- **STS (Security Token Service):** een STS is een systeem dat authenticatie en autorisatie afzondert van de andere componenten van een softwaresysteem. Zo kan de STS gebruikt worden door meerdere systemen en hoeven de applicaties zelf niet constant met de beveiliging rekening te houden. Het gebruik van een STS wordt verder toegelicht bij 9.9.2 HET GEBRUIK VAN EEN SECURITY TOKEN SERVICE.
- **XML (eXtensible Markup Language):** net zoals bij JSON kunnen gegevens naar deze vorm omgezet worden om over het internet te verzenden. XML heeft een striktere opmaak dan JSON en vraagt meer tekens om dezelfde gegevens te verzenden dan bij JSON.

## 9.6 Architectuur

Een degelijke mobiele applicatie heeft ook nood aan degelijke architectuur. Voor de eindgebruiker is dit in principe niet belangrijk, maar het vereenvoudigt wel de realisatie van enkele functies die voor de eindgebruiker wel belangrijk zijn. Zo moet de applicatie op elk moment kunnen reageren op acties van de gebruiker. Het inladen van data gebeurt daarom best in de achtergrond.

Een andere vereiste bij deze applicatie was dat er zoveel mogelijk code gedeeld kan worden over Android, iOS en Windows Phone. Om dit te verwezenlijken werd de code opgedeeld in de projecten beschreven op Tabel 11.

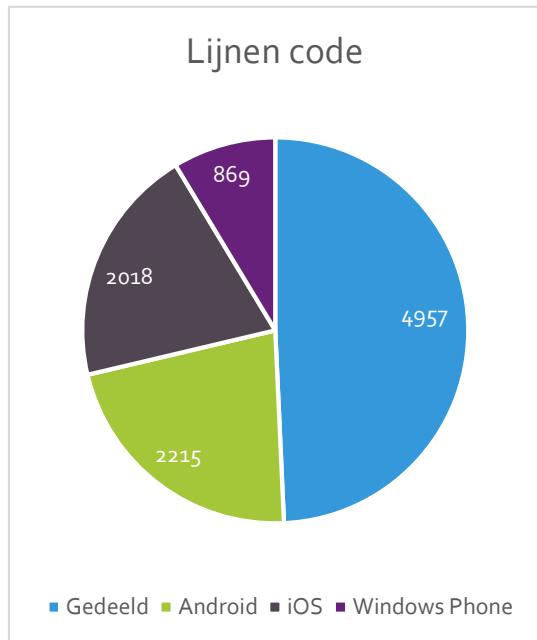
Project	Type	Doel
EducationApp	Portable Class Library	Code die gedeeld wordt over de andere projecten
EducationApp.Droid	Android app	App voor Android smartphones
EducationApp.iOS	iOS app	App voor iPhone
EducationApp.WinPhone	Windows Phone app	App voor Windows Phone
EducationApp.UnitTests	Unit tests	Unit tests van de gedeelde code
EducationApp.Metrics	Console app	Meet aandeel gedeelde code

Tabel 11: structuur van de applicatie(s)

Het doel was dus om zoveel mogelijk code in het gedeelde project te plaatsen en alle cruciale logica uit de app-projecten te houden. Het resultaat aan het einde van de stage is zichtbaar in Tabel 12.

app	totaal lijnen code	unieke lijnen code	gedeelde lijnen code	unieke code	gedeelde code
Android	7172	2215	4957	30,88%	69,12%
iOS	6975	2018	4957	28,93%	71,07%
Windows Phone	5826	869	4957	14,92%	85,08%
Gemiddeld	6658	1701	4957	24,91%	75,09%

Tabel 12: hoeveelheid gedeelde code



Figuur 20: aantal lijnen code per project

### 9.6.1 MVVM Light Toolkit

Om het resultaat in Tabel 12 te realiseren, werd gebruik gemaakt van een bestaand MVVM framework. Het aanbod aan MVVM frameworks is groot, maar de keuze viel voor MVVM Light Toolkit ([9.15.2.15 MVVM LIGHT TOOLKIT](#)) omdat het al jaren veel succes blijkt te hebben bij Windows Phone apps en geen denkpatronen opdringt.

Het framework bevat de volgende hulpmiddelen die elk optioneel te gebruiken zijn:

- Superklasse voor ViewModel en Model: vereenvoudigt het werken met het observable-observerpatroon (zie [9.7.3 OBSERVABLE-OBSERVER](#) voor verduidelijking).
- Navigatie: service om cross-platform te navigeren tussen meerdere views
- Superklasse voor commando's: vereenvoudigt het werken met het commandpatroon (toegelicht in [9.7.5 COMMAND](#)).

MAND)

- Dispatcher helper: deze helperklasse zorgt ervoor dat code eenvoudiger op de thread uitgevoerd kan worden die gebruikt wordt door de user interface (UI-thread). Dit is nodig om race condities, deadlocks en problemen met geheugentoegang te vermijden.
- Data binding framework: dit zorgt voor de koppeling tussen ViewModel en View

Bij data binding worden eigenschappen van een view gekoppeld aan gegevens in een view model. Wanneer een van beide verandert, zorgt het binding framework ervoor dat de andere van de twee ook aangepast wordt.

De API's voor Windows Phone bevatten een ingebouwd binding framework, maar voor Android en iOS bestaat dit niet. MVVM Light biedt daarvoor een eigen binding framework aan. Uiteindelijk bleek het een slechte keuze te zijn om hiervan gebruik te maken aangezien dit vaak de oorzaak van bugs bleek te zijn. Zo gebeurde het vaak dat de binding stopte met werken nadat nieuwe gegevens ingeladen waren.

### 9.6.2 Gedeelde code

De volgende onderdelen van de code van de apps konden gedeeld worden:

- Models: alle klassen die gegevens bijhouden afkomstig van de web API
- View models: de view models houden gegevens bij zodat deze op een eenvoudige manier in een view gebruikt kunnen worden (zie verder 9.7.1 MODEL-VIEW-VIEWMODEL (MVVM))
- View model locator: de *locator* bevat verwijzingen naar alle view models
- Abstracties van services: alle interfaces die bepalen welke functies de *services* hebben.
- Services: services bieden elk enkele functies aan die samen horen zoals het ophalen van gegevens van de web API, authenticatie, vertalingen... Heel wat services konden dezelfde implementatie voor de drie platformen gebruiken.

De volgende onderdelen werden voor elke app apart ontwikkeld:

- Views: lay-out en opmaak van een scherm
- Services met platformspecifieke code: *services* waarvan de implementatie verschilt per platform. Dit gaat dan om services die zorgen voor authenticatie, vertalingen en integratie met andere toepassingen op het toestel (browser, e-mail...).

### 9.6.3 Windows Phone app

Zoals blijkt uit Tabel 12 moest voor de Windows Phone app het minste extra code geschreven worden. De verklaring hiervoor is dat het .NET-platform waarop de app gebouwd werd, eigen is aan Windows Phone. Daarnaast bevatten de Windows Phone API's heel wat functies die de ontwikkeling van een app vereenvoudigen en gemakkelijker zijn in gebruik. De Windows Phone app bestaat dan ook voornamelijk uit schermen die de visuele lay-out van de app definiëren en enkele hulpmiddelen zoals *converters* die objecten van het ene formaat naar het andere omzetten.

### 9.6.4 Android app

De Android app bevat voornamelijk de volgende onderdelen:

- Activities: een *activity* is een volledig scherm in een app dat opgebouwd kan zijn uit verschillende andere, herbruikbare onderdelen.
- Fragments: een *fragment* is een stukje visuele opmaak dat opnieuw kan gebruikt worden in meerdere activities.
- Menus, toolbars, listitems: dit zijn hulpmiddelen die de opmaak bepalen van kleine stukjes uit het design die vaak opnieuw gebruikt worden.

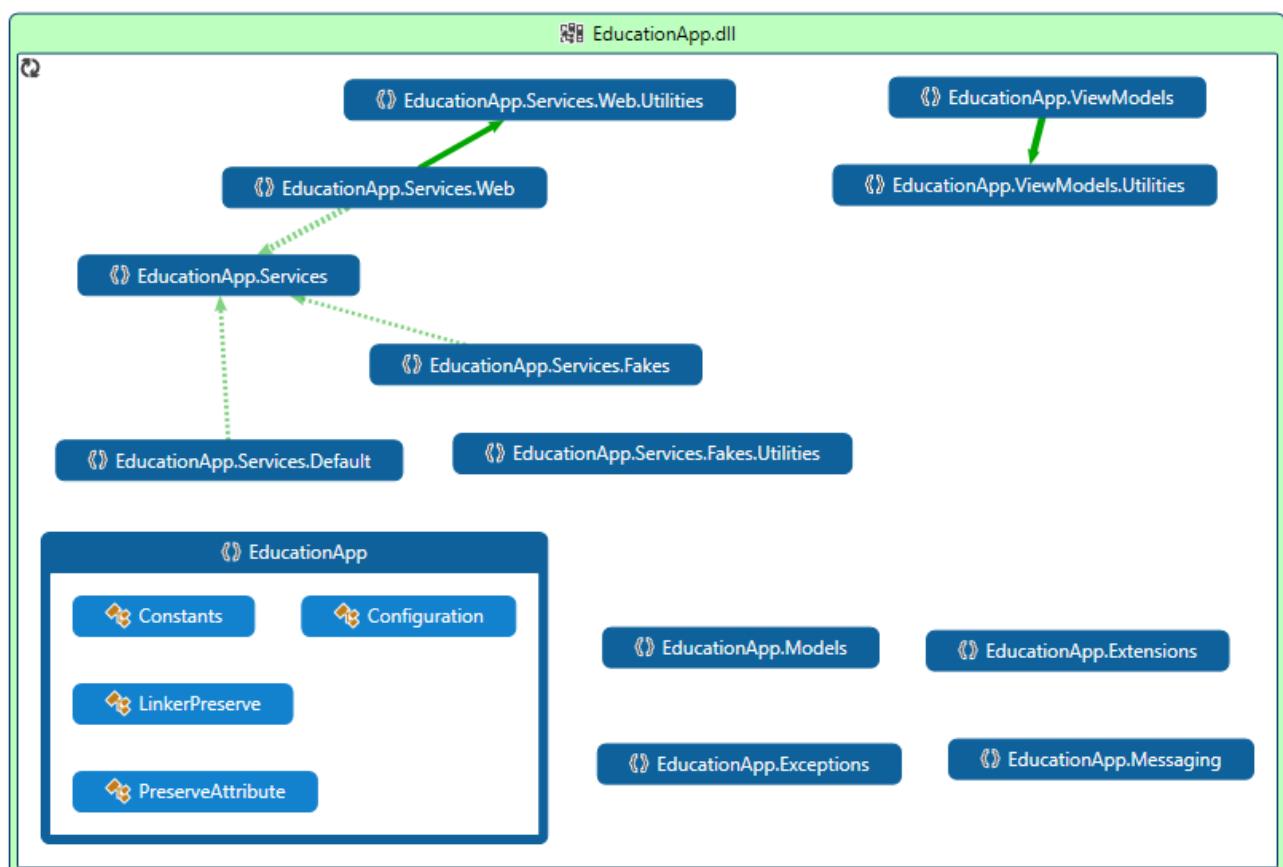
### 9.6.5 iOS app

De iOS app bevat voornamelijk de volgende onderdelen:

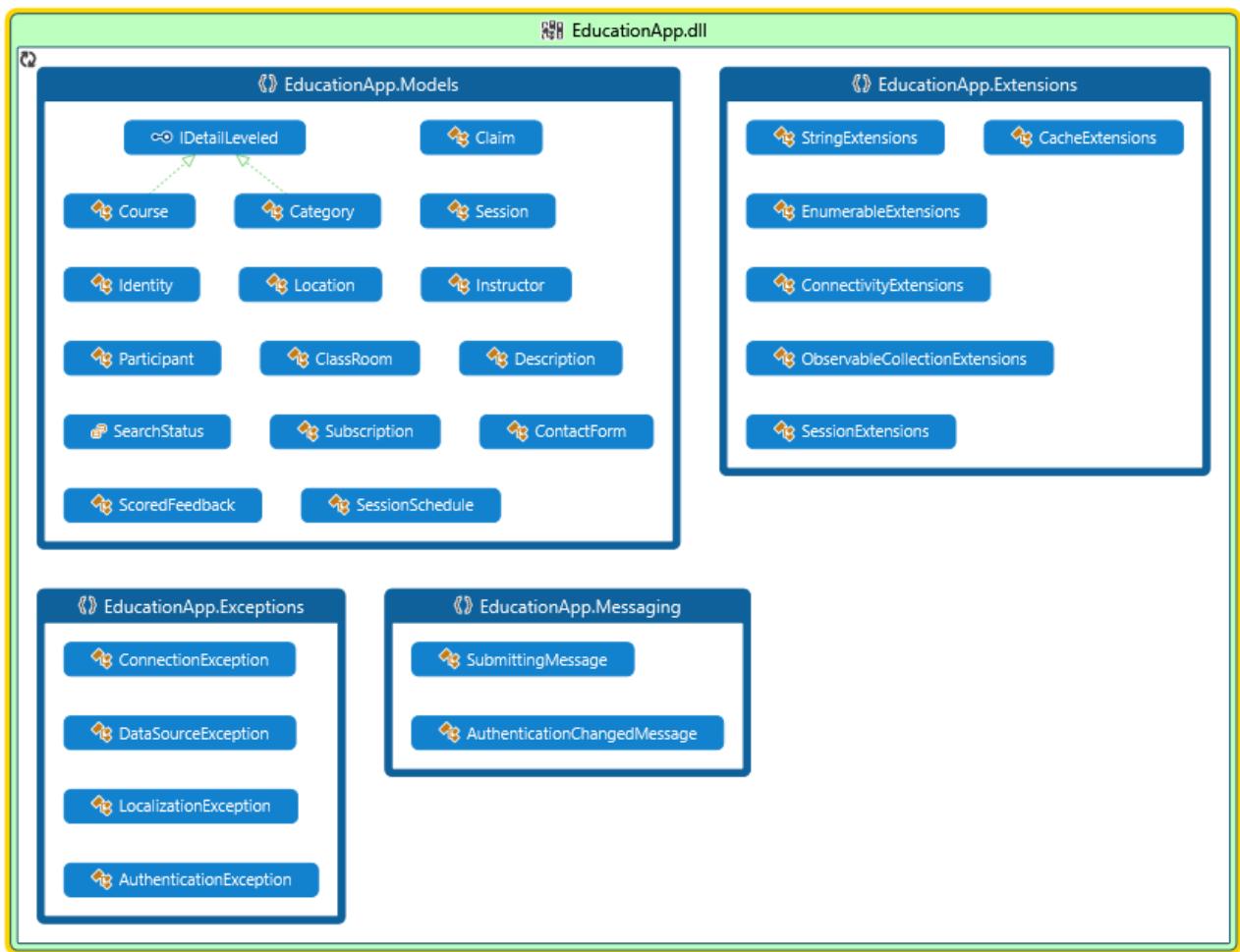
- Main.Storyboard: dit *storyboard* bevat alle schermen van de applicatie. Er hoeft weinig tot geen code geschreven te worden om een app visueel te ontwerpen voor iOS. De ontwikkelaar heeft de keuze om het design in code of in dergelijke storyboards te definiëren.
- ViewControllers: een *view controller* bevat de code die bepaalt hoe een scherm uit het storyboard functioneert. Het bevat code om knoppen en links te koppelen, design interactief aan te passen op basis van gegevens of acties van de gebruiker...

### 9.6.6 Overzicht

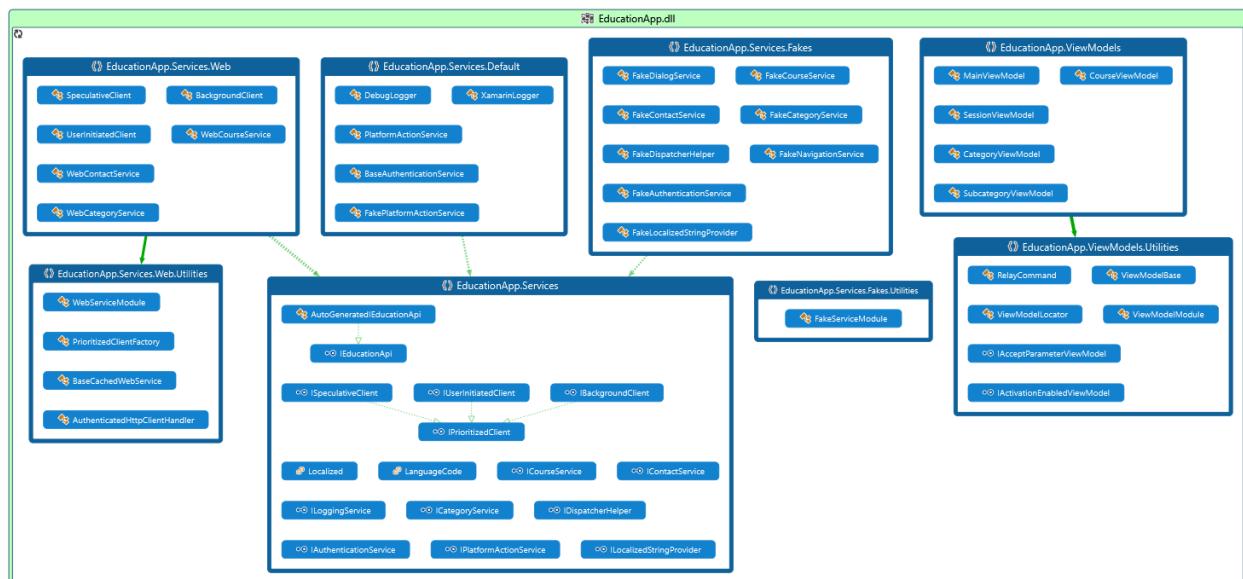
Onderstaande klassendiagrammen geven een overzicht van de structuur die beschreven werd in de voorstaande alinea's.



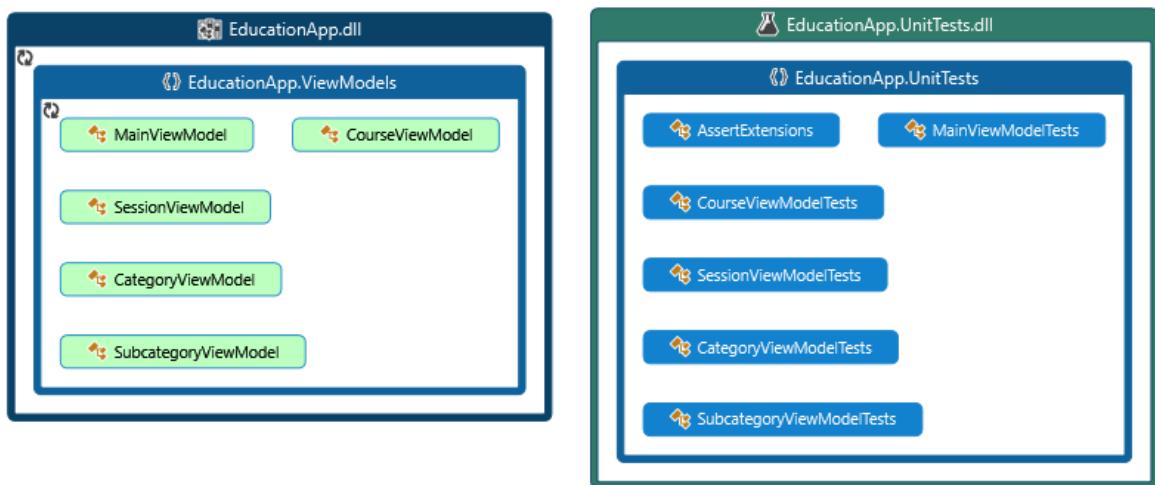
Figuur 21: overzicht van het project met de gedeelde code



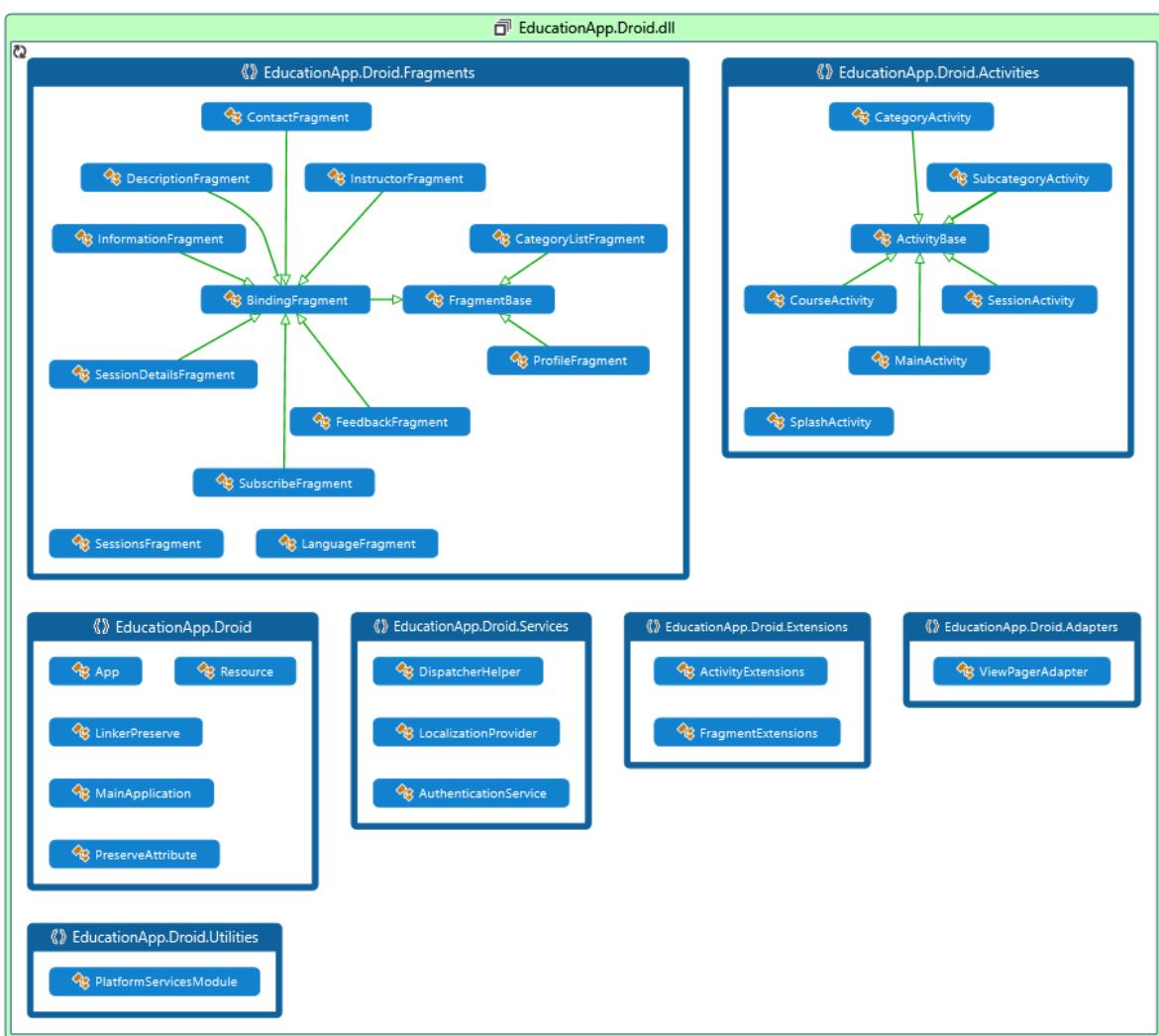
Figuur 22: statische gegevens (geen business logica) in de gedeelde code



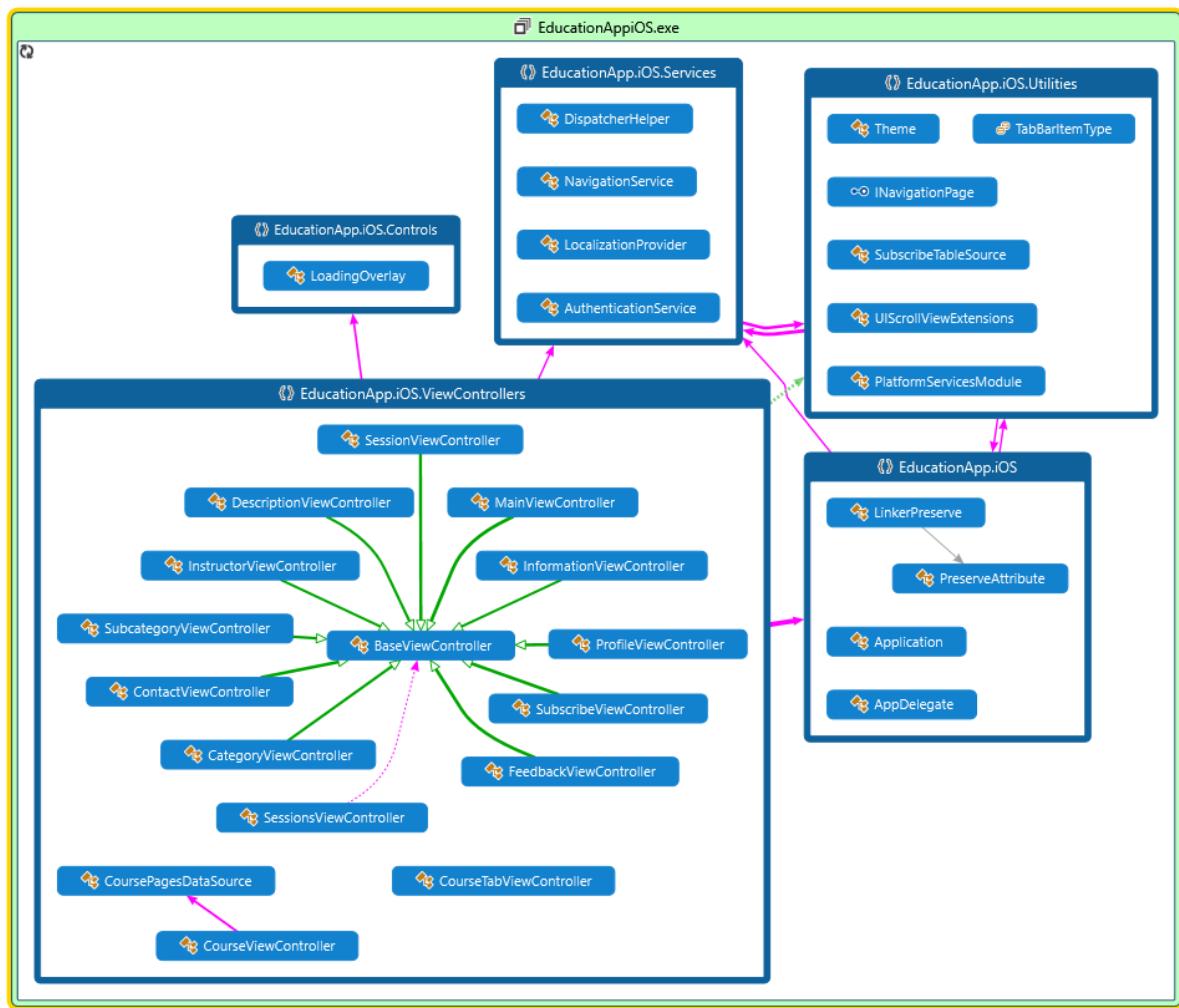
Figuur 23: business logica in de gedeelde code (view models en services)



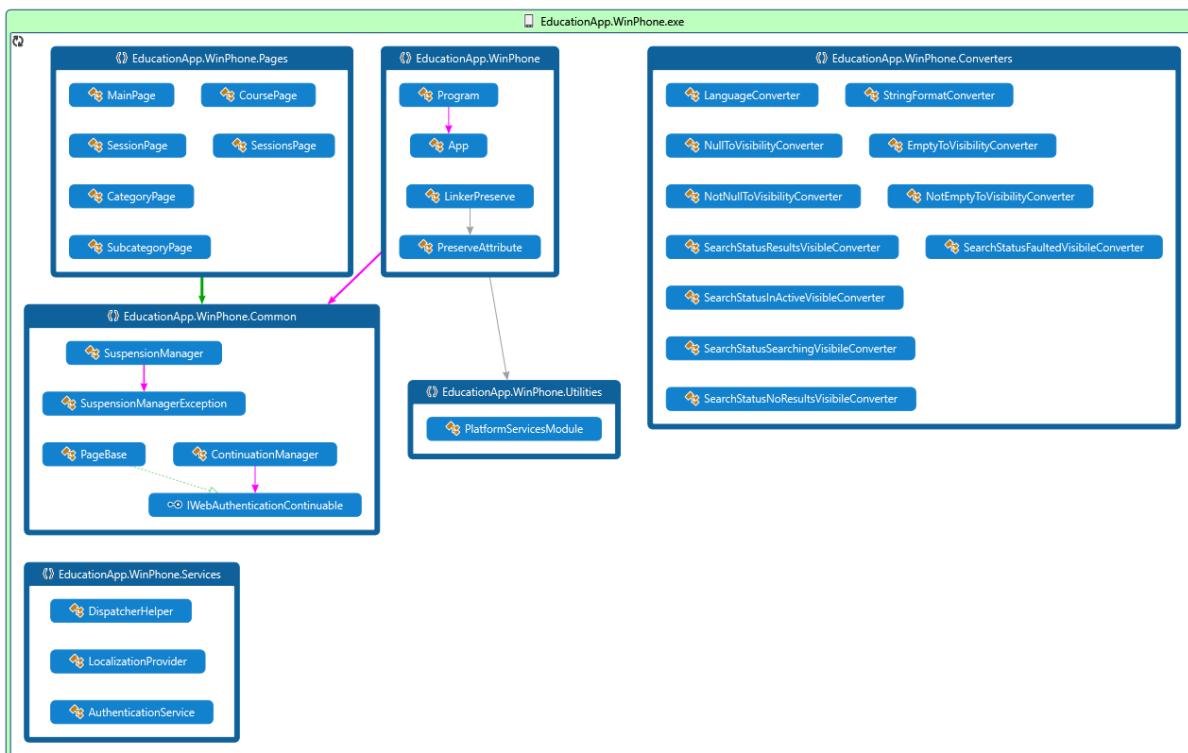
Figuur 24: view models met hun respectievelijke unit tests



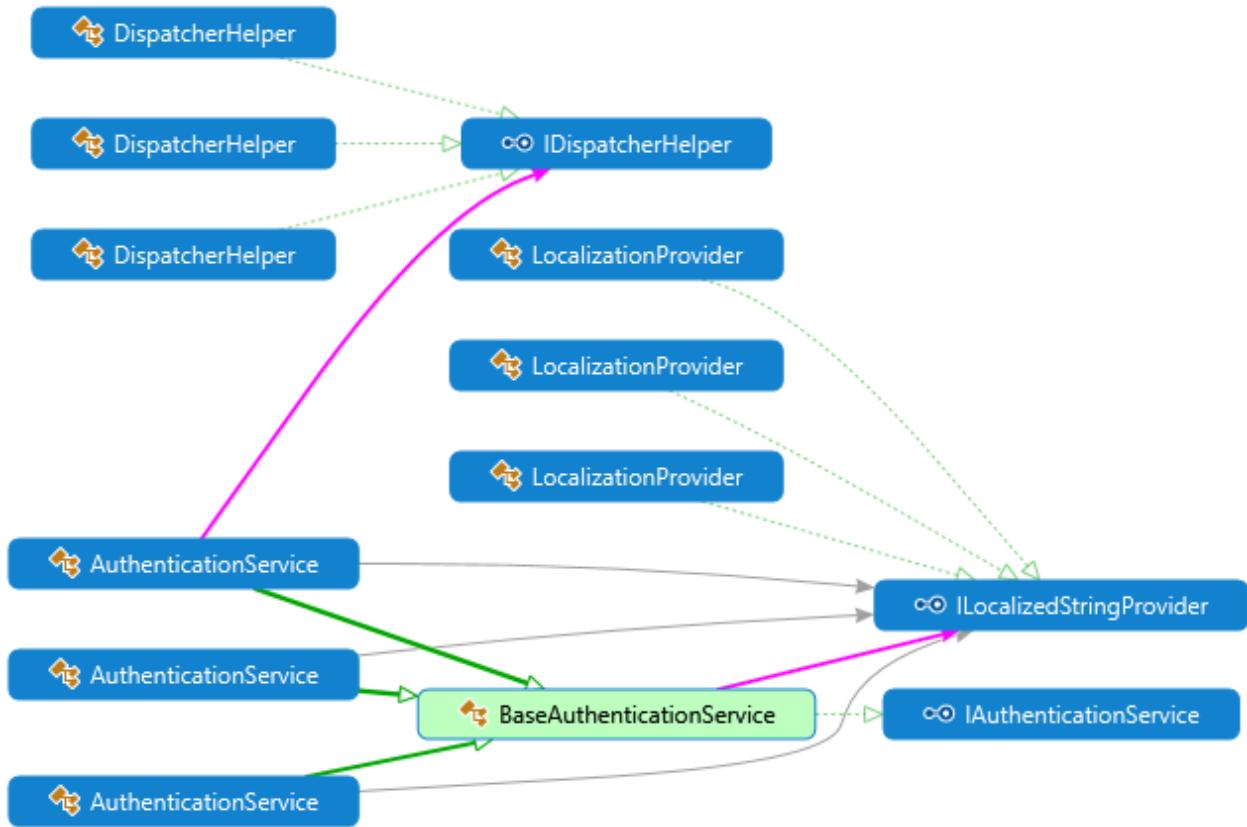
Figuur 25: overzicht van de code van de Android app



Figuur 26: overzicht van de code van de iOS app



Figuur 27: overzicht van de code van de Windows Phone app



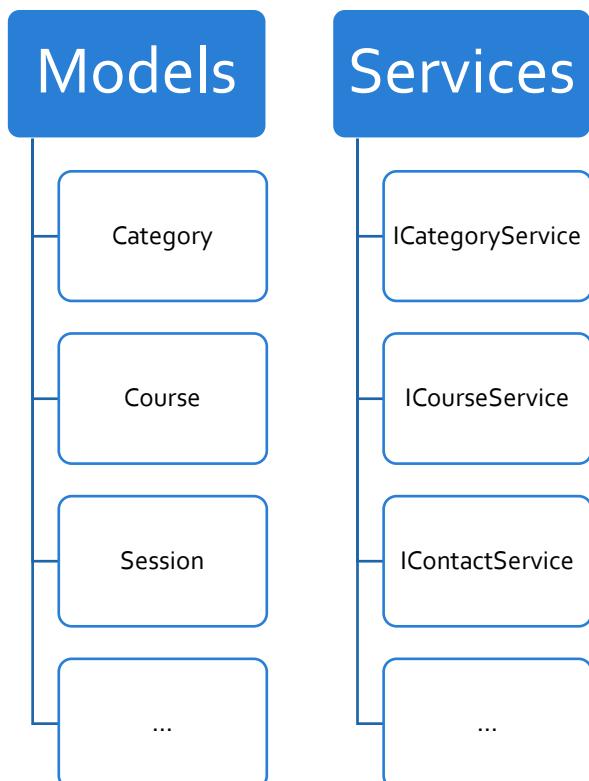
Figuur 28: platformspecifieke services

## 9.7 Gebruikte design patronen

### 9.7.1 Model-View-ViewModel (MVVM)

MVVM is een variatie op het bekende Model-View-Controller-patroon (MVC) dat vaak bij mobiele of desktopapplicaties gebruikt wordt. De verantwoordelijkheden van de componenten in de applicatie worden opgesplitst om duidelijker code te creëren. Zoals de naam al zegt, bestaat dit patroon uit drie onderdelen.

#### 9.7.1.1 Model



Figuur 29: inhoud van Model bij MVVM

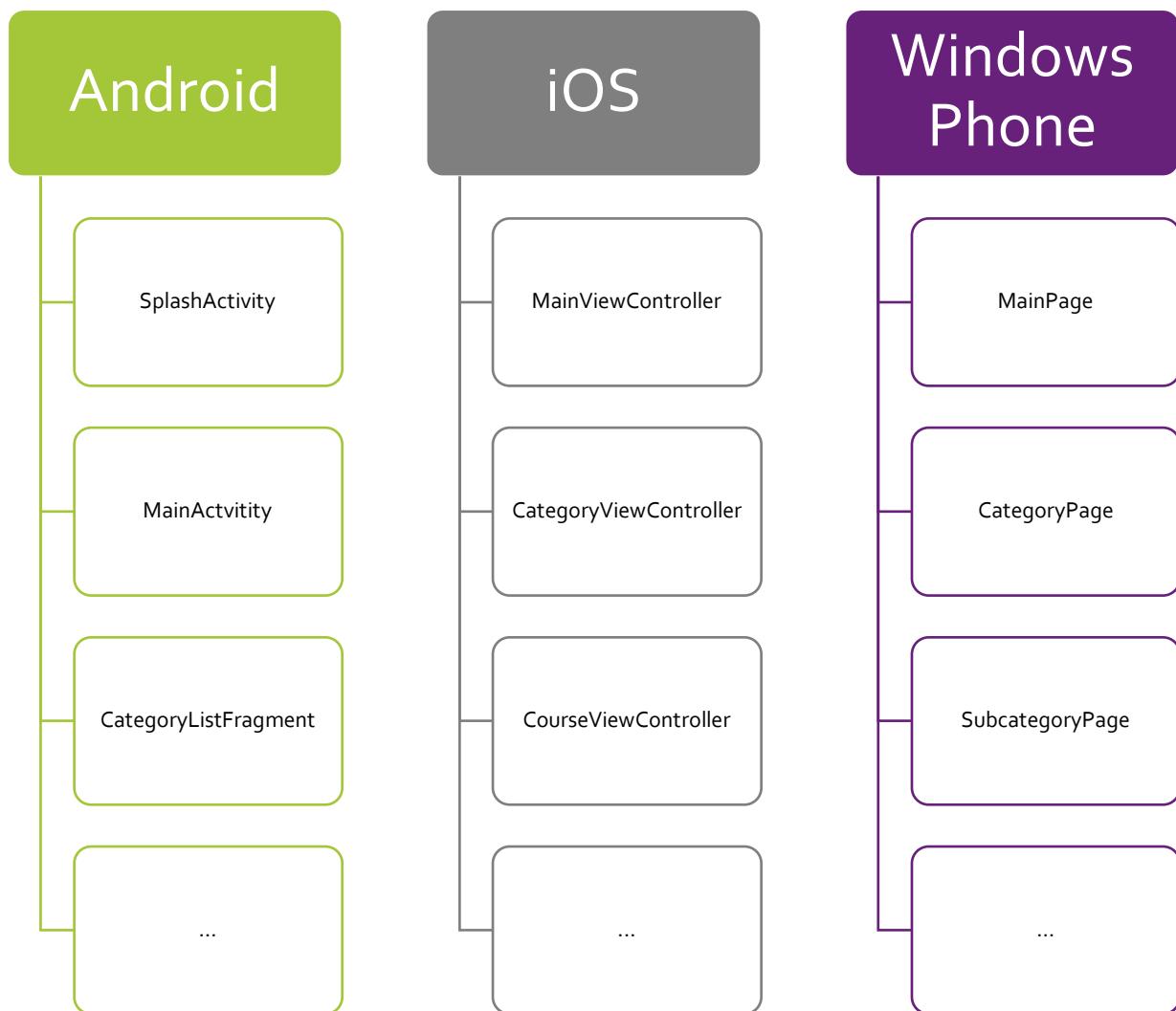
Model bevat alle gegevens die de applicatie nodig heeft en de logica om deze gegevens op te halen, te verzenden of te verwerken. Dit onderdeel kan later herbruikt worden wanneer een tweede applicatie van dezelfde gegevens gebruik wil maken. Het model bestaat uit service-klassen die nodig zijn om gegevens op te halen en model-klassen die nodig zijn om gegevens te structureren.

Bij de RealDolmen Education app bevatte dit onder andere de klassen in Figuur 29. Bij een Xamarin app kunnen alle klassen uit de Model-component volledig gedeeld worden over de drie apps.

#### 9.7.1.2 View

De View-component bevat alle code en lay-out-bestanden die nodig zijn voor de visuele weergaves van de applicatie. Deze klassen kunnen ook logica bevatten, maar deze mag enkel gerelateerd zijn aan de visuele weergave en mag geen enkele invloed hebben op de gegevens.

De views zijn afhankelijk van het mobiele platform waarvoor ze ontworpen zijn. Ze kunnen niet gedeeld worden bij een Xamarin app. Er is geen directe relatie tussen de gegevens uit de Model-component en de views. De hoeveelheid views hangt af van hoe de gegevens weergegeven worden, maar gegevens kunnen soms op meerdere views weergegeven worden en één view kan heel wat klassen uit de gegevens weergeven. Figuur 30 geeft enkele voorbeelden van views uit de app voor RealDolmen Education.



Figuur 30: inhoud van View bij MVVM

#### 9.7.1.3 ViewModel

De ViewModel-component van MVVM is de lijm tussen de models en de views. Een ViewModel bevat logica die bepaalt welke functies een applicatie aanbiedt. Het gaat dan om de manier waarop de gegevens beschikbaar gesteld worden (als lijst, een enkel gegeven, zonder details, met details) en om welke acties uitgevoerd kunnen worden. Deze laatste worden normaliter aangeboden als zogenaamde *commando's*.

De view models kunnen gedeeld worden over de mobiele platformen heen bij een Xamarin app aangezien de app op elk mobiel platform dezelfde functies aanbiedt. Figuur 31 geeft enkele voorbeelden uit de Real-Dolmen Education app weer.

MainViewModel	CategoryViewModel	SubcategoryViewModel
<ul style="list-style-type: none"> <li>• SearchStatus</li> <li>• SearchValue</li> <li>• Categories</li> <li>• FoundCourses</li> <li>• ShowCategoryDetailsCommand</li> <li>• ShowCourseDetailsCommand</li> <li>• Activate</li> </ul>	<ul style="list-style-type: none"> <li>• Category</li> <li>• ShowDetailsCommand</li> <li>• Activate</li> </ul>	<ul style="list-style-type: none"> <li>• Category</li> <li>• ShowCourseDetailsCommand</li> <li>• Activate</li> </ul>

Figuur 31: ViewModel-component bij MVVM

### 9.7.2 Dependency Inversion

Het SOLID-principe (Martin, 2000) bevat vijf principes die softwareontwikkelaars helpen bij het ontwerpen van een betere klassenstructuur in code. Inversion of Control zorgt voor de realisatie van het volledige SOLID-concept met nadruk op het vijfde principe: dependency inversion. SOLID behartigt de mogelijkheden om code duidelijk te structureren, eenvoudig te veranderen of uit te breiden en makkelijk te onderhouden.

Dependency inversion raadt programmeurs aan om niet af te hangen van concrete implementaties, maar van abstracties. In code wordt dus aangeraden om klassen afhankelijk te maken van interfaces in plaats van van andere klassen. Dit zorgt ervoor dat de gebruikte implementatie van de interface later eenvoudig kan uitgewisseld worden door een andere.

Een voorbeeld uit de RealDolmen Education app maakt dit wat duidelijker. De applicatie maakt gebruik van de dienst Xamarin.Insights om exceptions, crashes en andere foutmeldingen bij te houden in een online logboek. Een ontwikkelaar zou op elke plaats in code waar hij of zij iets wilt loggen kunnen verwijzen naar Xamarin.Insights. Wanneer dan later besloten wordt om van Xamarin.Insights over te stappen naar de gelijkaardige dienst HockeyApp, zou het veel werk kosten om de verwijzingen naar Xamarin.Insights een voor een aan te passen naar een gelijkaardige functie van HockeyApp.

Bovenstaand voorbeeld valt eenvoudig om te vormen naar een variant waarbij dependency inversion wel behartigd wordt. Er wordt een interface `ILoggingService` opgesteld die net zoals Xamarin.Insights aanbiedt om exceptions en dergelijke te loggen. Op plaatsen waar iets gelogd moet worden, wordt dan deze `ILoggingService` opgeroepen in plaats van Xamarin.Insights. Dan rest de ontwikkelaar nog het voorzien van een implementatie van Xamarin.Insights voor deze `ILoggingService`, bijvoorbeeld `InsightsLoggingService`. Tot slot moet de `InsightsLoggingService` nog ingevuld worden waar een `ILoggingService` gevraagd wordt. Later kan dan een `HockeyAppLoggingService` opgesteld worden die de `ILoggingService` implementeert en zijn de aan te passen verwijzingen miniem.

De nodige implementaties voor een abstractie kunnen ingevuld worden dankzij een Inversion-of-Control-container (IoC). Deze container bevat registraties en houdt zo bij welke implementatie ingevuld moet worden op elke plaats waar een bepaalde abstractie gevraagd wordt. Dit kan op drie manieren: constructor injection, property injection en injectie met een service locator.

#### 9.7.2.1 *Constructor injection*

De constructor van een klasse vraagt als parameters alle abstracties waarvan de klasse afhankelijk is. De IoC-container kan deze automatisch invullen en zo een nieuwe instantie van de klasse genereren.

#### 9.7.2.2 *Property injection*

Soms is constructor injection onmogelijk en kan beroep gedaan worden op property injection. Hierbij overloopt de IoC-container alle publieke properties met setters van een klasse en worden waar mogelijk implementaties ingevuld die overeenkomen met de registraties in de IoC-container.

#### 9.7.2.3 *Service Locator*

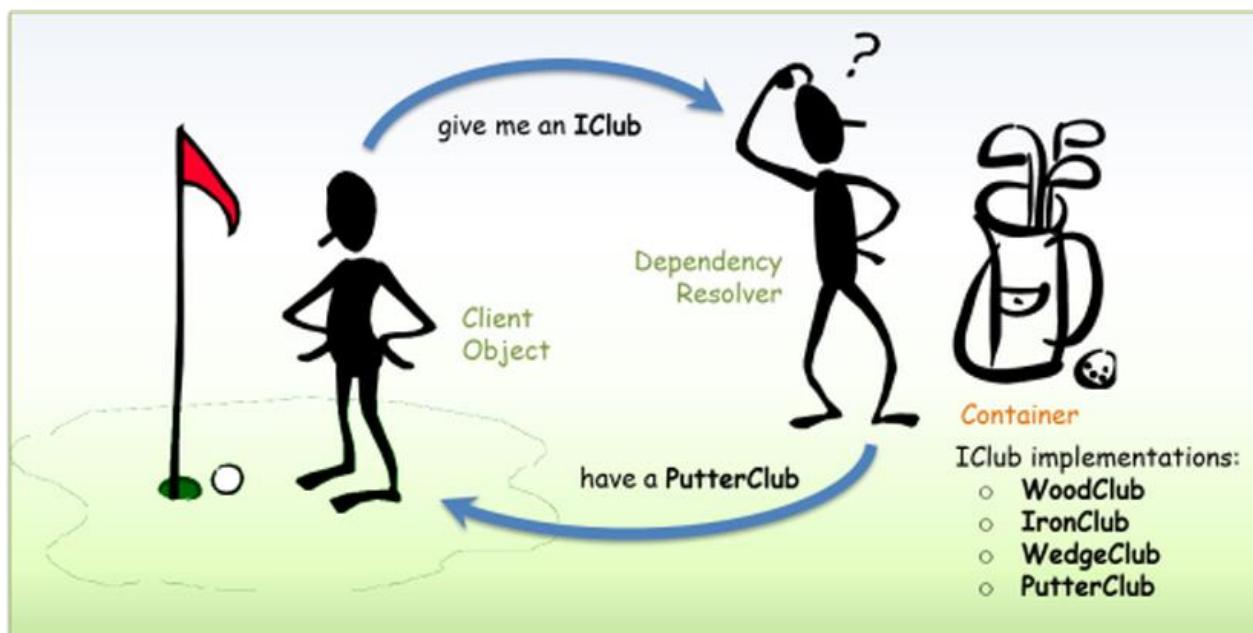
Wanneer het gebruik van een IoC-container niet mogelijk is, kan een service locator gebruikt worden. Deze locator is beschikbaar via een statische klasse en kan op elke plaats in de code opgeroepen worden. Dankzij een koppeling met de IoC-container kan de service locator ad hoc een implementatie voor een gevraagde abstractie voorzien.

#### 9.7.2.4 Toepassing

Er bestaan heel wat populaire dependency injection frameworks zoals Ninject, Autofac, Prism... Bij de RealDolmen Education app werd gekozen voor Autofac (9.15.2.2 AUTOFAC) dat iets meer mogelijkheden biedt dan de concurrentie en gemakkelijk te koppelen valt met Microsofts Service Locator (9.15.2.5 COMMON SERVICE LOCATOR).

Dit patroon werd veelvuldig gebruikt doorheen de volledige code van de RealDolmen Education app. Code die niet gedeeld kan worden over de verschillende mobiele platformen, kan hiermee beperkt worden. Wanneer een platformspecifieke implementatie noodzakelijk is, kan deze implementatie geïnjecteerd worden en hoeft de gedeelde code enkel te verwijzen naar de abstractie.

Ook bij design-time data kunnen zo de registraties uitgewisseld worden. Tijdens het visueel ontwerpen van de apps wordt design-time data weergegeven in plaats van gegevens die van de web API opgehaald worden. De IoC-container vult automatisch een implementatie voor de web API of een implementatie voor design-time data in wanneer nodig.

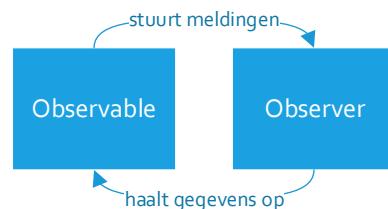


Figuur 32: voorbeeld van dependency injection (Web Camps Team, 2013)

#### 9.7.3 Observable-Observer

Dit patroon bestaat uit twee delen die met elkaar interageren. Enerzijds is er de observable-component. Dit bestaat uit een verzameling van gegevens die de eigenschap hebben om mogelijks te veranderen. De observable kan meldingen uitsuren wanneer er gegevens gewijzigd zijn. Een observer kan nadien zich abonneren op deze meldingen. Zo blijft de observer op de hoogte van de laatste gegevens die de observable bevat. Elk observable kan meerdere observers hebben. Dit wordt verduidelijkt in Figuur 33.

Dit patroon is cruciaal voor een degelijke mobiele applicatie. Om het scherm van een applicatie op elk moment te laten reageren op acties van de gebruiker, mag de applicatie niet blijven wachten op externe gegevensbronnen. In de plaats hiervan, kan dit patroon gebruikt worden. Een scherm abonneert zich op meldingen van de gegevensbron en past zichzelf aan wanneer de gegevens gewijzigd zijn. In tussentijd kan het scherm blijven reageren op handelingen van de gebruiker.



Figuur 33: werking van observable-observer patroon

Dit gaat hand in hand met het MVVM-patroon (9.7.1 MODEL-VIEW-VIEWMODEL (MVVM)). Het ViewModel stuurt meldingen naar de View wanneer de inhoud ervan veranderd is.

#### 9.7.4 Builder

Het builder-patroon wordt gebruikt om een uniforme manier aan te bieden om gelijkaardige soorten objecten aan te maken. Het werd gebruikt bij het aanmaken van de juiste HttpClient (9.8.3 TECHNISCHE WERKING). Een methode is in staat om HttpClient-objecten aan te maken die elk op een andere manier reageren, afhankelijk van de prioriteit die aan de builder gegeven wordt.

#### 9.7.5 Command

Het command-patroon schermt acties af. In een scherm kunnen dergelijke commando's aangeroepen worden die elk dezelfde methoden aanbieden:

- `CanExecute()`: geeft aan of dit commando op dit moment uitgevoerd kan worden
- `Execute()`: voert het commando uit

De schermen die de commando's oproepen bevinden zich in de platformspecifieke code, maar de inhoud en werking van het commando wordt bijgehouden in de gedeelde code.

#### 9.7.6 Overige

In de app voor RealDolmen Education werden nog andere bekende ontwerppatronen gebruikt, maar ze zijn het niet waard om individueel overlopen te worden aangezien ze weinig impact hadden op de architectuur of werking van de app:

- Singleton
- Chain of Responsibility
- Template Method

## 9.8 Caching

### 9.8.1 Frequentie problemen bij mobiele applicaties

Een van de requirements van de RealDolmen Education app was dat de app gedeeltelijk offline moest kunnen werken. Een mobiel toestel heeft niet altijd een (stabiele of snelle) internetverbinding en hier dient dus rekening mee gehouden te worden bij het creëren van de app.

De oplossing hiervoor is caching. Categorieën, lessen en andere gegevens worden eenmalig opgehaald en nadien op de smartphone zelf opgeslagen in een soort van buffer of cache. Er zijn enkele feiten die in deze context belangrijk zijn:

- De gebruiker van de applicatie moet zelf zo weinig mogelijk merken van het al dan niet beschikken over een internetverbinding.
- Mobiele internetverbindingen zijn vaak trager dan een Wi-Fi-verbinding.
- Mobiele internetverbindingen hebben vaak een beperking op de bandbreedte die ze maandelijks mogen verbruiken. Dit is vooral het geval bij internationale reizigers.
- Een gebruiker schakelt soms bewust zijn/haar internetverbinding uit. Er is niet altijd een internetverbinding beschikbaar.

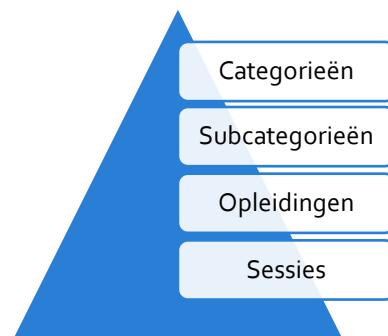
### 9.8.2 Werking van de cache

De RealDolmen Education app houdt hier op de volgende manier rekening mee:

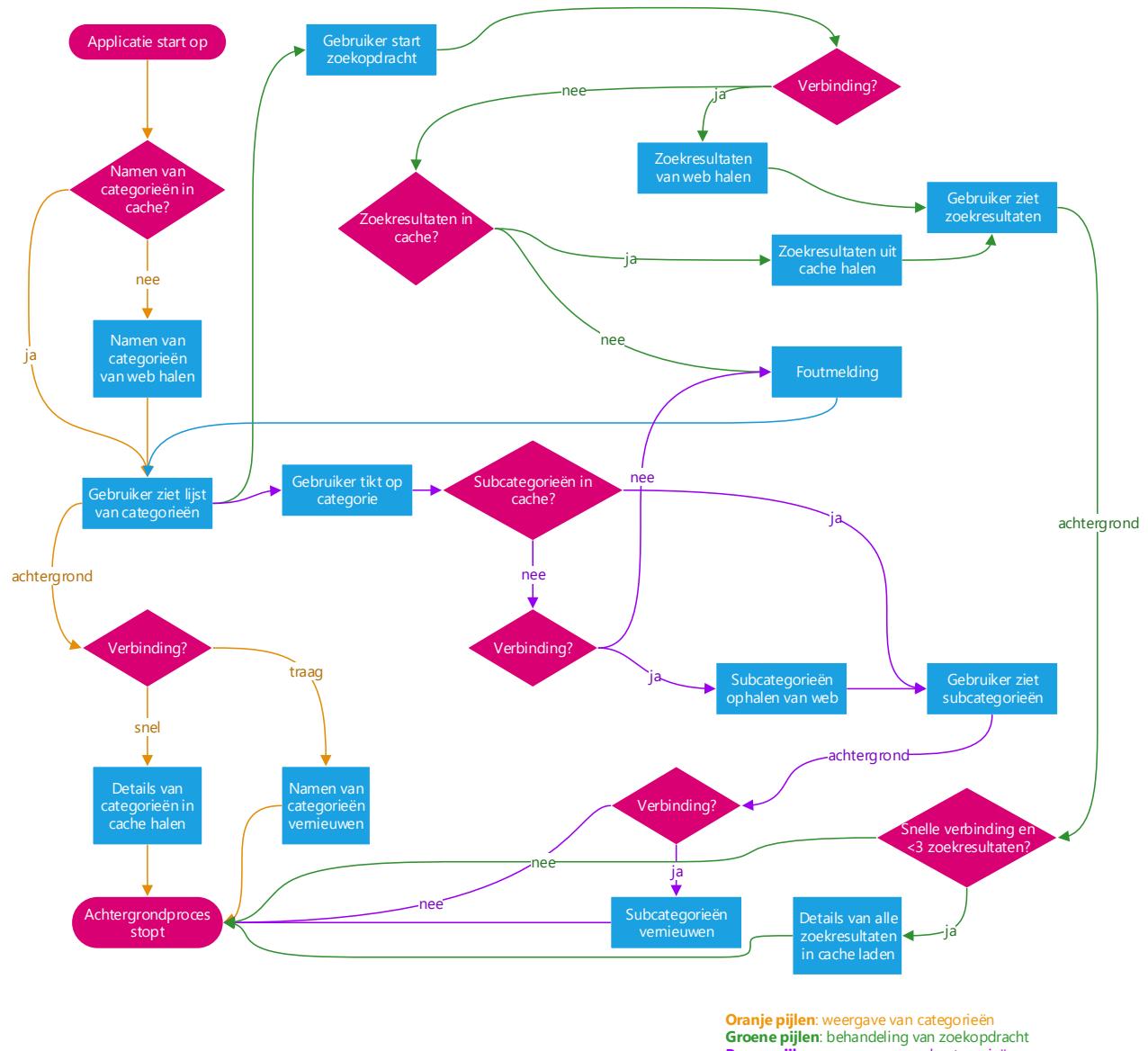
- ✓ Alle gegevens die ingeladen worden, blijven 10 dagen beschikbaar op het toestel zelf. Dit aantal dagen kan eenvoudig aangepast worden en hoeft niet hetzelfde te zijn voor alle gegevens (bv. de hoofdcategorieën 50 dagen en de opleidingen zelf slechts 3 dagen).
- ✓ De applicatie detecteert of de gebruiker een vermoedelijk snelle internetverbinding heeft of niet. Hier wordt gekeken naar het type verbinding: 3G- en 2G-verbindingen worden als vermoedelijk trage verbindingen beschouwd en alle andere types zijn vermoedelijk snelle verbindingen.
- ✓ Bij een snelle internetverbinding worden gegevens proactief/speculatief ingeladen. Bij een trage internetverbinding worden enkel de gegevens ingeladen die de gebruiker met 100% zekerheid te zien zal krijgen.

Om de werking van de cache te doorgronden is het interessant om het gegevensmodel van RealDolmen Education in Figuur 34 te bekijken.

In Figuur 35 wordt vervolgens weergegeven hoe de cache gebruikt wordt in de eerste fases van de applicatie. De werking van de cache is analoog in de diepere lagen van het gegevensmodel (opleidingen en sessies). Om ervoor te zorgen dat de app steeds responsief blijft, worden gegevens indien mogelijk op de achtergrond ingeladen.



Figuur 34: gegevensmodel van RealDolmen Education



Figuur 35: gebruik van cache in de eerste fases van de applicatie

### 9.8.3 Technische werking

Om dit technisch te verwezenlijken, werd er van enkele NuGet packages gebruik gemaakt:

- Akavache (meer informatie over mogelijkheden en werking: 9.15.2.1 AKAVACHE)
- Fusillade (meer informatie over mogelijkheden en werking: 9.15.2.7 FUSILLADE)
- Modern HTTP Client (meer informatie over mogelijkheden en werking: 9.15.2.13 MODERNHTTPCLIENT)

Alle gegevens over categorieën, opleidingen, sessies enz. worden lokaal in een SQLite-gegevensbank opgeslagen. Het beheer hiervan gebeurt volledig door Akavache.

Wanneer gegevens niet in de cache beschikbaar zijn, moeten ze van het web opgehaald worden. Hiervoor wordt gebruik gemaakt van een zogenaamde *HttpClient*. Om dit proces te versnellen werd GZIP-compressie ingeschakeld en werd de ingebouwde *HttpClient* vervangen door Modern HTTP Client die heel wat sneller werkt.

In Figuur 35 is te zien hoe gegevens soms op de achtergrond ingeladen worden en soms op de voorgrond. Logischerwijze zijn de gegevensoverdrachten op de voorgrond veel belangrijker en moeten deze voorrang krijgen om de wachttijd van de gebruiker te beperken. Hier zorgt Fusillade voor.

Akavache werkt zelfstandig, maar de andere componenten moeten natuurlijk met elkaar interageren om een correct geconfigureerd object te instantiëren dat gegevens van het web kan ophalen. Onderstaande code bevat documentatie die verklaart hoe het instantiëren verloopt. De methode GetClient wordt aangeroepen met een prioriteit en vervolgens wordt het *Factory* ontwerppatroon toegepast.

```
public static class PrioritizedClientFactory
{
    private static readonly Uri BaseAddress;

    static PrioritizedClientFactory()
    {
        BaseAddress = new Uri(Configuration.ApiUri);
    }

    // Priority is ofwel speculatief/proactief ofwel voorgrond
    public static Lazy<IEducationApi> GetClient(Priority priority)
    {
        // handler van Modern HTTP Client wordt aangemaakt
        var nativeHandler = new NativeMessageHandler();
        if (nativeHandler.SupportsAutomaticDecompression)
        {
            // inschakelen van compressie (GZip)
            nativeHandler.AutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;
        }

        // handler van Fusillade wordt aangemaakt met de aangegeven prioriteit, gebruikt
        // onderliggend de handler van Modern HTTP Client
        var rateLimitedHandler = new RateLimitedHttpMessageHandler(nativeHandler, priority);

        // eigen handler die authenticatie regelt wordt aangemaakt, gebruikt onderlig-
        // gend de handler van Fusillade
        var authEnabledHandler = new AuthenticatedHttpClientHandler(() => BaseAuthenticationService.GetAuthorizationHeaderAsync(), rateLimitedHandler);

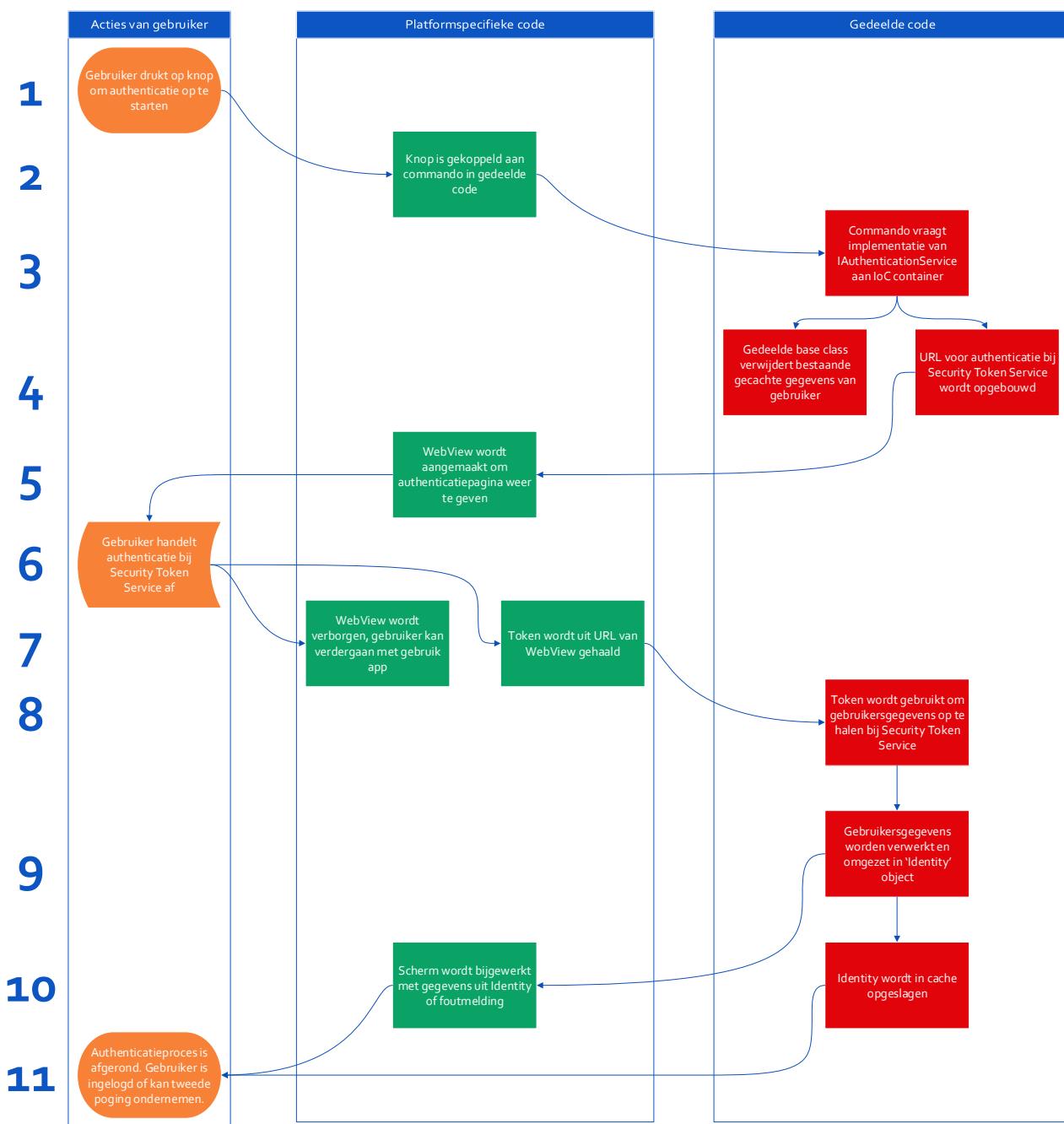
        // HttpClient met de authenticatiehandler wordt aangemaakt en ingesteld
        var rateLimitedClient = new HttpClient(authEnabledHandler)
        {
            BaseAddress = BaseAddress
        };

        // service die met API communiceert wordt aangemaakt, gebruikt de HttpClient
        return new Lazy<IEducationApi>(() => RestService.For<IEducationApi>(rateLimitedClient));
    }
}
```

Alle code die zorgt voor het ophalen en cachen van gegevens kan gedeeld worden over de drie mobiele platformen.

## 9.9 Authenticatie

Tijdens de voorbereidingsfase van dit project werd bij de technische analyse al dieper ingegaan op de manier waarop authenticatie verloopt bij de RealDolmen Education app. Dit kan nu uitgebreid worden met de manier waarop de verschillende stukken code met elkaar interageren. Het protocol dat bij de authenticatie gebruikt wordt heet OAuth 2. Hier bestaan verschillende vormen van die elk een eigen 'flow' hebben. Deze flow bepaalt welke stappen nodig zijn om tot een succesvolle authenticatie te komen. De flow die hier gebruikt wordt is de *Implicit Flow*.



Figuur 36: verloop van authenticatie en verhouding gedeelde code

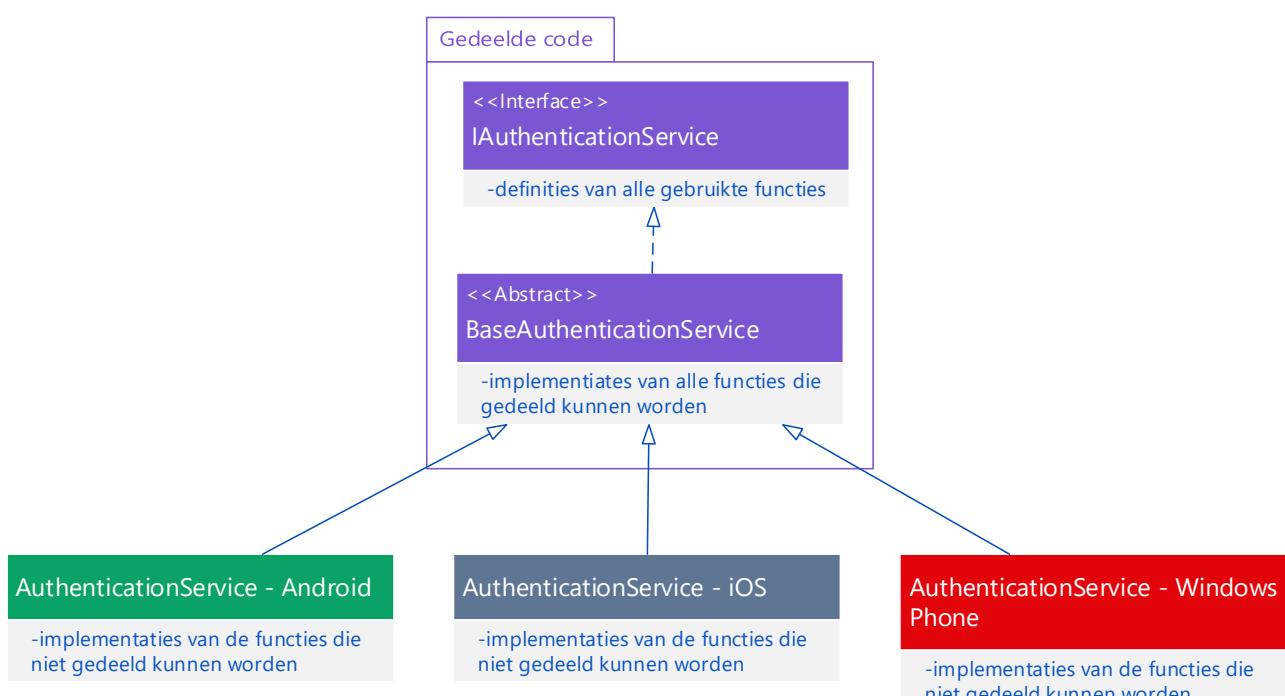
Figuur 36 geeft opnieuw het verloop van de authenticatie weer, maar geeft ook aan welke functies gebruik maken van code die over alle platformen gedeeld wordt en welke functies platformspecifieke code vereisen. Stappen die naast elkaar weergegeven worden op het diagram, worden op hetzelfde moment uitgevoerd in de applicatie.

Vanaf stap 7 kan de gebruiker de applicatie opnieuw bedienen terwijl de authenticatie in de achtergrond verder afgerond wordt. Wat hier ook opvalt is dat de meeste en de belangrijkste stappen telkens in de gedeelde code uitgevoerd worden. Aangezien dit ook de code is die het meeste getest wordt, verkleint dit de kans op bugs in de drie apps.

### 9.9.1 Structuur van de code

De code bevat de volgende interface en klassen:

- **IAuthenticationService** (gedeelde code): interface die alle methodes definieert die in het hele proces gebruikt worden
- **BaseAuthenticationService** (gedeelde code): abstracte klasse die de methodes uit de **IAuthenticationService** implementeert die gedeeld kunnen worden over de verschillende platformen
- **AuthenticationService** (Android / iOS / Windows Phone): implementaties van de functies die platformspecifieke code vereisen. Deze klasse erft over van de **BaseAuthenticationService**.



Figuur 37: hoe code gedeeld wordt bij authenticatie

### 9.9.2 Het gebruik van een Security Token Service

RealDolmen Education maakt voor authenticatie gebruik van een STS of Security Token Service. Dit scheidt de authenticatie en autorisatie af van de app. Dit betekent ook dat de STS door meerdere diensten gebruikt kan worden: een mobiele app, een website, een desktopprogramma...

De applicatie geeft aan de STS de volgende parameters door:

- **Redirect URI<sup>7</sup>**: de URI waarnaar de gebruiker moet teruggestuurd worden na voltooiing van de authenticatie. Wanneer de STS de gebruiker naar deze URI stuurt, is het aan het besturingssysteem van het toestel om de URI op te pikken en opnieuw de mobiele applicatie te laten zien. Deze URI verschilt per mobiel platform.

<sup>7</sup> Unique Resource Identifier: wordt vaak ook URL of hyperlink genoemd

- **Client ID:** een uniek ID dat aanduidt om welk type toepassing het gaat (bv. mobiele app of website).
- **Response type:** aan het response type valt af te leiden welke flow van OAuth 2 gebruikt wordt. Zo weet de STS hoe de applicatie de authenticatie opnieuw oppikt en verder afhandelt. Voor de *implicit flow* die hier gebruikt werd, is dit *id\_token token*.
- **Scope:** de scope duidt aan welke toegang de app nodig heeft. Zo heeft de app de scope *profile* nodig om de naam van de gebruiker op te mogen vragen. Andere scopes waren nodig om in naam van de gebruiker gegevens naar de web API te kunnen sturen.
- **Nonce:** number only used once. Dit is een willekeurig getal dat in de app gegenereerd wordt. Aan de hand van dit unieke nummer kunnen twee extra verificaties uitgevoerd worden. Elk getal mag slechts één keer<sup>8</sup> gebruikt worden voor authenticatie. Daarnaast wordt meestal aan het einde van authenticatie nagegaan of dit getal niet veranderd is tijdens de authenticatie. Als aan een van beide verificaties niet voldaan werd, dan is de authenticatie ongeldig en wordt het *access token* uit de STS onbruikbaar gemaakt.

Behalve de nonce zijn de parameters zowel ingesteld in de app zelf als in de STS. Deze moeten allemaal overeenkomen om veilige authenticatie te garanderen.



Figuur 38: vereenvoudigd verloop van *implicit flow*

Op basis van deze parameters, stelt de STS een loginpagina op. Nadat de gebruiker zich hierin aanmeldt, wordt hij/zij gevraagd of de app toegang krijgt tot persoonlijke gegevens. Wanneer ook dit goedgekeurd wordt, wordt de gebruiker naar de redirect URI gestuurd. Aan het einde van deze URI is een JWT of JSON Web Token en een *access token* toegevoegd.

Een JWT is een lange reeks karakters die gedecrypteerd kunnen worden naar JSON dat op zijn beurt ook weer uitgelezen kan worden. Dit bevat heel wat informatie, maar is bij de implicit flow minder nuttig. Wat hier wel uitgehaald kan worden is een uniek ID dat de gebruiker identificeert en onderscheidt van andere gebruikers. Dit ID is het enige<sup>9</sup> gegeven dat gegarandeerd uniek is voor elke gebruiker.

Het *access token* is het meest cruciale resultaat van de authenticatie. Deze karakterreeks dient als unieke en tijdelijke sleutel om vanuit de app met de web API te communiceren in naam van de gebruiker.

### 9.9.3 Windows Phone

Om platformspecifieke code af te handelen op Windows Phone, kan de [WebAuthenticationBroker](#) uit de Windows Phone API's gebruikt worden. Dit vormt een soort brug tussen de code van de app en de authenticatie in de Security Token Service.

De authenticatie-URL die opgebouwd wordt in de gedeelde code, wordt doorgegeven aan deze broker. Die opent de URL dan in een browser en laat de gebruiker inloggen. Wanneer het inloggen voltooid is en de

<sup>8</sup> Meestal worden deze getallen ergens tijdelijk bijgehouden en na verloop van tijd weer verwijderd. Zo blijft het beveiligingsmechanisme werken en is de kans kleiner dat een gebruiker toevallig een getal hergebruikt dat al door iemand anders gebruikt werd.

<sup>9</sup> De STS kan zelf andere beperkingen opleggen zoals iedereen verplichten om een uniek e-mailadres te gebruiken, maar de mobiele app heeft hier geen weet van. Daarom zijn deze gegevens nooit gegarandeerd uniek vanuit het perspectief van de app.

applicatie weer getoond moet worden, kan de broker het JWT en het access token uit de URL halen en deze doorgeven aan de [BaseAuthenticationService](#) in de gedeelde code.

Om bij Windows Phone na het inloggen weer terug te keren naar de app, is een correcte Redirect-URI nodig. Dit is een unieke URI die de ontwikkelaar kan genereren a.d.h.v. de broker.

#### 9.9.4     Android en iOS

Voor de platformspecifieke code werd gebruik gemaakt van Xamarin.Auth (zie ook 9.15.1.2 XAMARIN.AUTH). Deze component bevat verschillende functies, maar de klasse die gebruikt werd is de [WebRedirectAuthenticator](#).

Hiermee is het mogelijk om op Android en iOS een WebView te laten zien met een pagina naar keuze en de WebView weer te sluiten wanneer een bepaalde URL (de Redirect URI) gedetecteerd wordt.

Ook hier kunnen nadien eenvoudig de nodige resulterende tokens uit de URI gehaald worden.

## 9.10 Logging

Wanneer een mobiele app afgeleverd wordt, is het belangrijk om de kwaliteit en stabiliteit van de app te blijven opvolgen. De kans dat een gebruiker van de app namelijk zelf de moeite neemt om problemen en crashes te melden is vrij klein. Het is waarschijnlijker dat de gebruiker de app van zijn smartphone verwijdert.

Het is daarom aan het ontwikkelteam om proactief te werk te gaan en problemen meteen op te lossen wanneer ze zich voordoen. Hierdoor is het essentieel dat de ontwikkelaars op de hoogte gehouden worden van problemen, crashes en andere informatie die hier zinvol kan zijn. Bij Xamarin apps liggen vaak *exceptions* aan de oorzaak van een crash. Het is dus belangrijk om te weten waar en wanneer exceptions zich voordoen en alle informatie hierrond te verzamelen (bv. stack trace).

Er bestaan verschillende manieren om dit aan te pakken en een ervan is *Xamarin.Insights* (9.15.2.19 XAMARIN.INSIGHTS). *Xamarin.Insights* biedt 3 krachtige hulpmiddelen aan:

- **Report:** informatie over exceptions, crashes...
- **Track:** bijhouden welke mogelijkheden van de app het meeste benut worden, waar gebruikers het vaakst op drukken, welke pagina's gebruikers het vaakst bezoeken...
- **Identify:** nagaan welke gebruikers het meeste actief zijn, welke gebruikers een bepaalde functie het vaakst benutten of wie het vaakst met crashes te maken heeft

Bij aanvang van de stage, was Report gratis. Track en Identify vielen buiten de scope van het project en waren dus de prijs niet waard in dit geval. Na de overname van Xamarin door Microsoft werd *Xamarin.Insights* samengevoegd met Microsofts HockeyApp dat gelijkaardige diensten aanbood.

*Xamarin.Insights* Report bestaat zelf ook uit verschillende mogelijkheden:

- In code kunnen exceptions manueel gerapporteerd worden.
- Onbehandelde exceptions (die vaak crashes veroorzaken) worden automatisch gerapporteerd.
- Bij een crash: gegevens over de crash en de exception die de crash veroorzaakt heeft worden op het toestel zelf opgeslagen. Wanneer de app de volgende keer geopend wordt, worden de opgeslagen gegevens naar *Xamarin.Insights* verzonden.
- Bij elke exception wordt een volledige stack trace en, indien wenselijk, extra informatie verzameld.
- Bij elk gegevenspunt worden ook gegevens vermeld over het toestel waarop de exception of crash zich voordeed:
  - Besturingssysteem en versie
  - Merk en model van het toestel
  - Versie van de app waarbij het probleem zich voordeed

Bovenstaande gegevens worden automatisch verzonden naar een e-mailadres en kunnen online geraadpleegd worden in een dashboard.

The screenshot shows the Xamarin Insights dashboard for the 'EducationApp.Testing' project. At the top left, a teal box displays 'Crash-free users PAST 30 DAYS' at 89.5%. To its right, a purple box shows 'Latest reports' with a 'LATEST CRASH' from 6 days ago and a 'LATEST WARNING' from 1 week ago. Below these are search and filter fields ('Search issues...', 'All Versions', 'All Time'). The main area lists '56 Unsymbolicated Crashes' with columns for 'Last Occurred', 'Count', 'Users Affected', and 'Status'. Three specific crashes are listed:

Category	Description	Last Occurred	Count	Users Affected	Status
DataSourceException	There was a problem retrieving the requested objects.	6 days ago	1	1 1.1%	OPEN
InvalidOperationException	The AppCompatDispatcherHelper cannot be called. Make sure that your main Activity derives from AppCompatActivityBase. (JimBobBennett.MvvmLight.AppCompat)	1 week ago	1	1 1.1%	OPEN
Exception	The application called an interface that was marshalled for a different thread. The application called an interface that was marshalled for a different thread.	1 week ago	1	1 1.1%	OPEN

Figuur 39: dashboard van Xamarin.Insights

This screenshot shows a detailed view of a single exception report. At the top, it displays the exception type 'DataSourceException' with the message 'There was a problem retrieving the requested objects.' It includes details like 'LAST OCCURRED 6 days ago', 'COUNT 1', and 'USERS Affected 1 1.1%' with an 'Open' button. Below this are four summary cards: 'Occurrences per Day' (0, 1, 1), 'Top Affected Version' (1.0.0.0, 100%), 'Affected OS' (WindowsPhone 8.1, 100%), and 'Affected Devices' (Microsoft Virtual, 100%). The main content area shows a stack trace for a guest user (ID 27928) on April 25, 2016, at 7:55am UTC. The user 'microsoft virtual WindowsPhone 8.1' crashed after 52s. The stack trace shows the exception was thrown by 'EducationApp.Exceptions.DataSourceException' with the message 'There was a problem retrieving the requested objects.' The stack trace also includes frames for 'EducationApp.Services.Web.WebCourseService.<FetchCourseDetailsAsync>d\_\_17.MoveNext()' and 'System.ObjectDisposedException'.

Figuur 40: voorbeeld van exception op Xamarin.Insights

Zoals duidelijk blijkt uit Figuur 39 en Figuur 40 werd Xamarin.Insights tijdens de stage succesvol geïntegreerd en is het zeker een aan te raden hulpmiddel voor elke Xamarin app. Dankzij de manier waarop Xamarin.Insights beschikbaar gesteld wordt (als NuGet package), is het eveneens mogelijk om dit bij Windows Phone apps te gebruiken, ook al komt er geen Xamarin aan te pas.

## 9.11 Vertalingen

Een van de requirements van de RealDolmen Education app is dat de app beschikbaar is in het Nederlands, het Frans en het Engels. Vanzelfsprekend is het zinloos om de vertalingen voor elk mobiel platform opnieuw te schrijven. De applicatie moet overal exact dezelfde functies hebben en alhoewel de lay-out hierdoor kan verschillen, mogen de omschrijvingen van de functies niet verschillen.

### 9.11.1 Vertalingen bij Android

Elk Android-project bevat een map genaamd *resources*. Hierin wordt alles bijgehouden dat geen pure code is, waaronder vertalingen. Deze bevinden zich in het bestand *strings.xml* in de submap *values*. Dit XML-bestand bevat een verzameling teksten (*strings*) die in de code of in de lay-out-bestanden opgeroepen kunnen worden. Elke tekstje wordt geïdentificeerd door een zogenaamde key:

```
<string name="Loading">Loading...</string>
```

Deze string kan vervolgens opgehaald worden met een functie:

```
Resources.GetText(Resource.String.Loading) // geeft Loading... terug
```

Om vervolgens ook teksten voor andere talen aan te maken, wordt de ISO-639-1-taalcode aan de naam van de map *values* toegevoegd:

- ❖ Android-project
  - resources
    - values
      - strings.xml – vertalingen in de standaardtaal (meestal Engels)
    - values-nl
      - strings.xml – vertalingen in het Nederlands
    - values-fr
      - strings.xml – vertalingen in het Frans

### 9.11.2 Vertalingen bij iOS

Terwijl bij Android en Windows Phone er een enkel taalbestand is per taal voor zowel teksten uit code als teksten uit lay-out, is dit bij iOS niet zo. iOS apps maken voor de lay-out vaak gebruik van zogenaamde *storyboards*. Dit zijn XML-bestanden die enkel in Apple XCode of de Xamarin Designer aangepast kunnen worden. Deze programma's geven de storyboards visueel weer en laten de ontwikkelaar toe om een lay-out visueel te definiëren in plaats van in code.

De structuur van taalbestanden in een Xamarin.iOS app ziet er als volgt uit:

- ❖ iOS-project
  - Base.Iproj
    - Localizable.strings – teksten voor de basistaal (meestal Engels)
  - resources
    - nl.Iproj
      - Localizable.strings – teksten in het Nederlands
      - StoryBoardX.strings – vertalingen voor teksten uit StoryBoardX in het Nederlands
    - fr.Iproj
      - Localizable.strings – teksten in het Frans
      - StoryBoardX.strings – vertalingen voor teksten uit StoryBoardX in het Frans

Voor de basistaal (Engels) moet er geen taalbestand voor de storyboards voorzien worden omdat deze teksten reeds in het storyboard zelf zijn opgeslagen.

De Localizable.strings-bestanden bevatten lijsten van vertalingen in de vorm *identifier* = "tekst", maar verder geen opmaak. Het zijn dus geen XML-bestanden zoals op de andere platformen. De storyboard-vertalingen bevatten verwijzingen naar de storyboards zelf. Deze verwijzingen gebeuren via automatisch gegenereerde identifiers die uit willekeurige tekens bestaan.

Vertalingen van storyboards gebeuren automatisch, vertalingen in code kunnen als volgt opgeroepen worden:

```
NSBundle.MainBundle.LocalizedString("Loading", comment) // geeft Loading... terug
```

De variabele *comment* is een eventuele hint om vertalers te helpen bij de vertalingen.

### 9.11.3 Vertalingen bij Windows Phone

De eenvoudigste manier om vertalingen te beheren, is terug te vinden bij apps voor Windows Phone. Windows Phone apps gebruiken XAML-bestanden voor hun design, een vorm van XML. Voor de vertalingen wordt er gebruik gemaakt van resource-bestanden. Elk visueel element kan een unieke identifier krijgen. Om vervolgens dit element te vertalen, wordt er in het resource-bestand verwezen naar de eigenschap van het visueel element. Onderstaand voorbeeld maakt dit wat concreter:

Een knop (*Button*) heeft een eigenschap *Content* die bepaalt welke tekst in de knop moet weergegeven worden. Er kan bv. een knop zijn om een e-mail te verzenden die dan als identifier *BtSendMail* heeft gekregen. In het resource-bestand wordt dan als identifier *BtSendMail.Content* gebruikt en als tekst "E-mail verzenden".

De structuur van de taalbestanden ziet er hier als volgt uit:

- ❖ Windows Phone-project

- Strings
  - en
    - Resources.resw
  - fr
    - Resources.resw
  - nl
    - Resources.resw

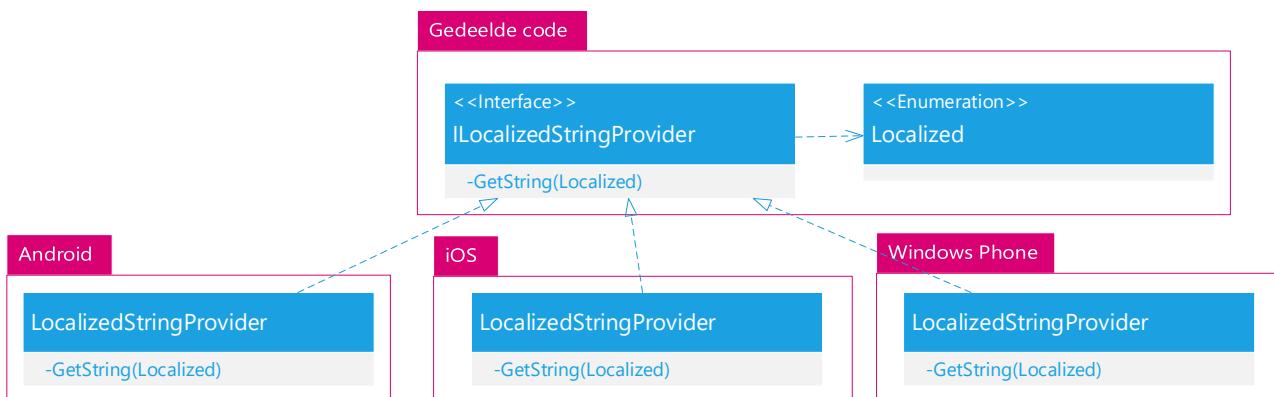
De resource-bestanden zijn XML-bestanden die eenvoudig aangepast kunnen worden in Microsoft Visual Studio. Om een vertaling in code op te halen, wordt een object van het type *ResourceLoader* aangemaakt die de methode *GetString(identifier)* aanbiedt.

### 9.11.4 Cross-platform vertalingen

De enige gelijkenissen tussen de vertalsystemen op de Android, iOS en Windows Phone zijn de ISO-639-taalcodes. Het is ook mogelijk om op elk platform de identifiers van bij Windows Phone te gebruiken. Om problemen te vermijden, moeten de punten (.) hierin vervangen worden door underscores (\_). Het blijft dan nog steeds een uitdaging om steeds op elk platform de juiste code te kunnen oproepen en om de vertalingen in de juiste vormen te gieten.

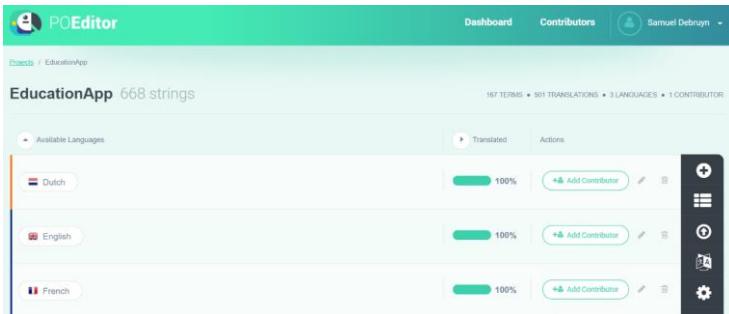
Om dit op te lossen werd gebruik gemaakt van dependency injection waar 9.7.2 DEPENDENCY INVERSION dieper op ingaat. Overal waar er vertalingen vereist zijn, wordt er verwezen naar een interface *ILocalizedStringProvider*. Deze interface heeft een methode die een vertaling teruggeeft en zelf als parameter een

waarde van de enumeratie `Localized` vraagt. Op elk platform is deze interface anders geïmplementeerd en de juiste implementatie wordt overal automatisch geïnjecteerd.



Figuur 41: implementaties voor vertalingen

Dit is slechts een deel van de oplossingen. Hierdoor kan de code om vertalingen op te roepen al gedeeld worden, maar de vertalingen zelf moeten nog omgezet kunnen worden naar het juist formaat op elk platform.



Figuur 42: overzicht van POEditor

standen in de juiste mappen te plaatsen. Ook dit zou eventueel verder nog geautomatiseerd kunnen worden met een PowerShell-script.

Om ook dit laatste obstakel van de baan te helpen, werd van de dienst POEditor.com gebruik gemaakt. Deze dienst kan vertalingen inlezen in elk van de eerder vernoemde formaten en nadien ook naar elk formaat exporteren. Daarnaast is het ook mogelijk om teksten automatisch te laten vertalen via een derde partij als Google Translate. Na het exporteren is het een kwestie van de punten door underscores te vervangen en de bestanden in de juiste mappen te plaatsen. Ook dit zou eventueel verder nog geautomatiseerd kunnen worden met een PowerShell-script.

## 9.12 Testing

Er bestaan verschillende manieren om software te testen alvorens deze vrij te geven. De meest voorkomende manieren zijn de volgende:

- **Acceptatietesten:** er wordt gekeken naar de functionele eisen van de klant, meestal in de vorm van user stories. Deze story wordt letterlijk geïnterpreteerd en uitgevoerd in de testversie van de software. Dit kan geautomatiseerd worden met een programma als Cucumber.
- **UI-testen:** een programma zoals Selenium kan automatisch klikken of andere acties uitvoeren in het te testen programma. Aan het einde van de test wordt gedetecteerd of een bepaalde afbeelding of tekst op het scherm staat om zo te bepalen of de test al dan niet geslaagd is.
- **Unit testen:** aan de hand van een test framework worden stukken code van de applicatie getest. Meestal komt elke klasse uit de code overeen met één testklasse en worden daarin per methode uit de geteste klasse verschillende testmethodes aangemaakt waarin wordt nagegaan of de methode uitvoert wat ervan verwacht wordt.

Voor dit project werd gekozen om de view models (zie ook 9.7.1 MODEL-VIEW-VIEWMODEL) te testen via unit tests. De redenen hiervoor zijn dat de view models gedeeld worden over de drie platformen en ze alle belangrijke logica bevatten die bepalen welke functies de app krijgt en hoe deze werken.

Een view model heeft voor heel wat functies afhankelijkheden op andere delen van de code. Omdat deze via interfaces gedefinieerd worden, kunnen de implementaties hiervan eenvoudig vervangen worden door testversies. Zo is het bijvoorbeeld belangrijk dat het `SessionViewModel` beroep doet op de methode `SendEmail` van een implementatie van de `IPlatformActionService`. Tijdens het testen is het dan niet nodig om na te kijken of er effectief een e-mail verzonden wordt door implementaties van de `IPlatformActionService`, maar wel belangrijk dat de `IPlatformActionService`. Er zijn implementaties voorzien van de `IPlatformActionService` die e-mails kunnen verzenden op Android, iOS en Windows Phone.

### 9.12.1 Mocking library: Moq

Het zou heel wat werk kosten moeten er ook implementaties gemaakt moeten worden van die afhankelijkheden voor de unit testen. Om dit te vermijden, kan beroep gedaan worden op een *mocking library* zoals Moq (meer details bij 9.15.2.14 Moq). Moq kan implementaties voorzien van een interface naar keuze en ad hoc het gedrag hiervan laten bepalen. Er kan ook worden geverifieerd hoeveel keer een methode van een interface aangesproken geweest is.

In onderstaand voorbeeld wordt getest of het view model `CategoryViewModel` kan navigeren naar de details van een subcategorie met behulp van de `INavigationService`.

```
[TestMethod]
    public void ShowDetails_Navigated_ValidCategory()
    {
        // Arrange
        var ns = new Mock<INavigationService>();
        ns.Setup(
            n => n.NavigateTo(Constants.Pages.SubcategoryDetailsKey, _fakeCategory.Sub-
categories.FirstOrDefault());
            _registrations.RegisterInstance(ns.Object);
        FinishRegistrations();

        // Act
       Vm.ShowDetailsCommand.Execute(_fakeCategory.Subcategories.FirstOrDefault());

        // Assert
        ns.Verify(
```

```
n => n.NavigateTo(Constants.Pages.SubcategoryDetailsKey, _fakeCategory.Subcategories.FirstOrDefault()),
    Times.Once);
}
```

Deze code maakt eerst een zogenaamde *mock* aan van de `INavigationService`. Er is dus geen concrete implementatie van nodig. In de volgende stap wordt bepaald hoe deze mock moet reageren als een bepaalde methode, hier `NavigateTo`, aangeroepen wordt. In dit geval verwacht de oproeper geen resultaat en werkt de methode zonder problemen.

In het tweede deel wordt de methode opgeroepen die getest moet worden, in dit geval gaat het over `ShowDetailsCommand.Execute` in het view model.

Tot slot wordt geverifieerd hoe vaak de methode `NavigateTo` aangeroepen werd. Als dit één keer aangeroepen werd, heeft de methode `ShowDetailsCommand.Execute` die we wilden testen zijn werk gedaan en is de test geslaagd.

### 9.12.2 Test coverage

Op deze manier werden alle functies van alle view models getest. Bij elke methode werden de afhankelijkheden bij elke test op een andere manier opgezet. Het is namelijk ook belangrijk om na te gaan hoe het view model reageert wanneer een van de afhankelijkheden (tijdelijk) niet correct werkt. Zo werd gepoogd om het bereik van de testen te maximaliseren. Hoeveel code er effectief getest werd, varieerde gedurende het project. Het uiteindelijke resultaat is zichtbaar in Figuur 43.

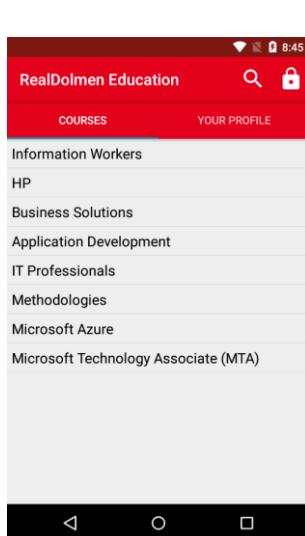
Symbol	Coverage (%)	Uncovered/TotalStmts.
▷ ⚡ EducationApp.Services.Web.Utilities	0%	77/77
▷ ⚡ EducationApp.Services.Web	0%	395/395
▷ ⚡ EducationApp.Services.Default	3%	131/135
▷ ⚡ EducationApp.Extensions	13%	132/152
▷ ⚡ EducationApp.Exceptions	32%	13/19
▷ ⚡ EducationApp.Services.Fakes	50%	92/183
▷ ⚡ EducationApp.Models	55%	365/814
▷ ⚡ EducationApp.ViewModels.Utilities	72%	32/116
◀ ⚡ EducationApp.ViewModels	76%	133/556
▷ 📂 CourseViewModel	58%	70/168
▷ 📂 SessionViewModel	66%	37/108
▷ 📂 CategoryViewModel	90%	7/71
▷ 📂 MainViewModel	90%	14/140
▷ 📂 SubcategoryViewModel	93%	5/69
▷ ⚡ EducationApp.Messaging	90%	1/10
▷ ⚡ EducationApp.Services.Fakes.Utilities	100%	0/4
▷ □ EducationApp.UnitTests	99%	4/642

Figuur 43: test coverage

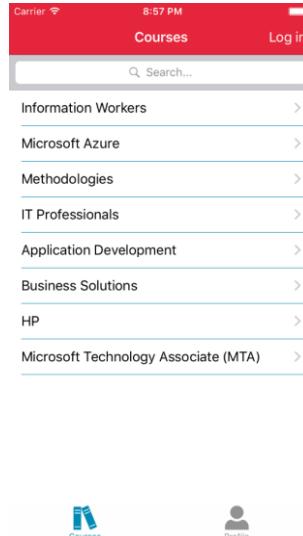
Met een dergelijk hoge *coverage* kan men ervan uitgaan dat mogelijke bugs hoogstwaarschijnlijk niet aan de view models te wijten zijn.

## 9.13 Visueel ontwerp

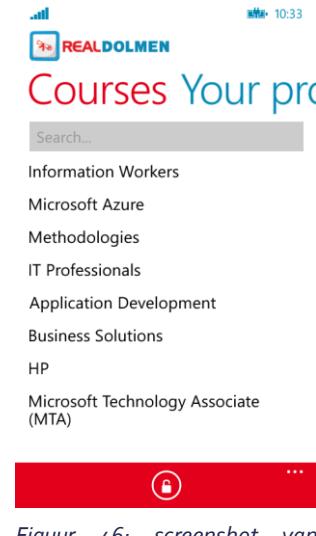
De eindgebruiker van een mobiele applicatie wordt het vaakst geconfronteerd met het visuele ontwerp van de applicatie. Een van de doelen van dit project was om code te delen over de mobiele platformen, maar de weergave op elk mobiel platform moet wel uniek zijn. Een belangrijk kenmerk hierbij is dat de trends en patronen worden gehanteerd die eigen zijn aan de mobiele platformen en aangeraden worden door respectievelijk Apple, Google en Microsoft.



Figuur 44: screenshot van hoofdscherm in de Android app



Figuur 45: screenshot van hoofdscherm in de iOS app

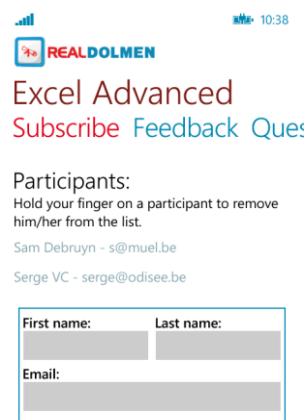


Figuur 46: screenshot van hoofdscherm in de Windows Phone app

In Figuur 44, Figuur 45 en Figuur 46 is telkens hetzelfde scherm te zien, maar voor een ander platform. Uit dit voorbeeld vallen heel wat zogenaamde patronen af te leiden die in de meeste applicaties op dat platform zitten. Zo moet de gebruiker niet nadenken over de werking van de app, maar zijn de functies meteen duidelijk. Enkel bij Android vallen er meerdere dimensies te

bespeuren. Bij Windows Phone vallen dan weer de strakke lijnen en eenvoud op. Typisch voor iOS zijn ook de tabknoppen onderaan het scherm. Dit in tegenstelling tot de tabs bovenaan bij Android en Windows Phone. Bij Windows Phone worden minder icoontjes gebruikt en de acties staan onderaan het scherm in plaats van bovenaan. Op Android is dan weer de zoekbalk verborgen onder een zoekknop terwijl deze bij iOS en Windows Phone altijd zichtbaar is. Een ontwikkelaar is altijd vrij over de specifieke implementatie, maar wordt toch aangeraden om de eerder beschreven patronen toe te passen.

Een mobiele applicatie biedt dankzij het aanraakscherm unieke mogelijkheden om een handige interface op te bouwen die vlotter werkt dan een desktop met toetsenbord en muis. In Figuur 47 zijn sliders te zien waarmee de gebruiker eenvoudig een score kan aanduiden op het feedbackformulier. Deze kunnen meteen versleept worden met een enkele aanraking.



Figuur 48: screenshot van inschrijfscherm op Windows Phone

Ook bij het inschrijfscherm kan het gebruik van de applicatie vereenvoudigd worden dankzij de mogelijkheden van een aanraakscherm. Dit scherm bevat een lijst van deelnemers. Om een deelnemer te verwijderen volstaat het om de naam van een deelnemer lang ingedrukt te houden.

### 9.13.1 View lifecycle management

Een app kan op veel manieren gebruikt worden en het valt zelden te voorspellen waar een gebruiker op zal drukken. Elk scherm kan tijdens het gebruik van



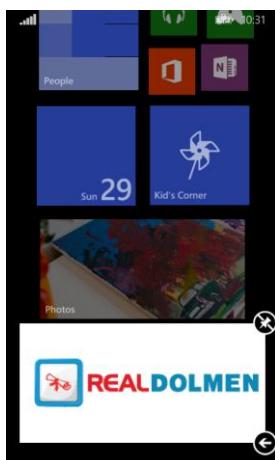
Figuur 47: screenshot van scherm om feedback te geven op Windows Phone

een applicatie meerdere keren weergegeven worden, al dan niet met andere informatie. Om een applicatie performant te houden moet nauwkeurig omgesprongen worden met het geheugenverbruik. Dit gebeurt op elk mobiel platform quasi volledig anders.

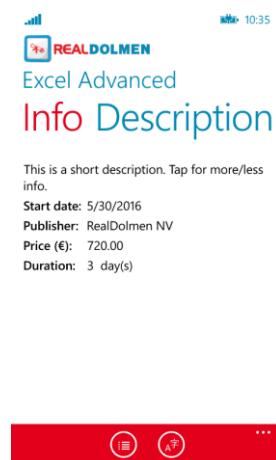
### 9.13.2 Windows Phone



Figuur 49: kleine tegel op Windows Phone



Figuur 50: brede tegel op Windows Phone



Figuur 51: blauwe statusbalk op Windows Phone



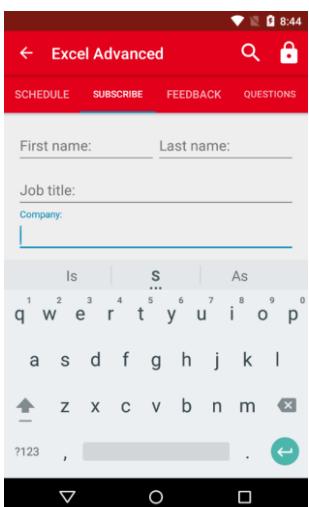
Figuur 52: vlakke elementen op Windows Phone

Microsofts mobiel platform biedt enkele unieke mogelijkheden om de branding van bedrijf in het platform te integreren. Zo kan het logo van de app een volledig andere afbeelding weergeven bij verschillende groottes van de 'tegel' om de app te openen. Dit is zichtbaar in Figuur 49 en Figuur 50. Ook de kleuren van de huisstijl kunnen verder geïntegreerd worden. In Figuur 51 is te zien hoe de statusbalk bovenaan het scherm aangepast is naar het blauw uit het logo van RealDolmen.

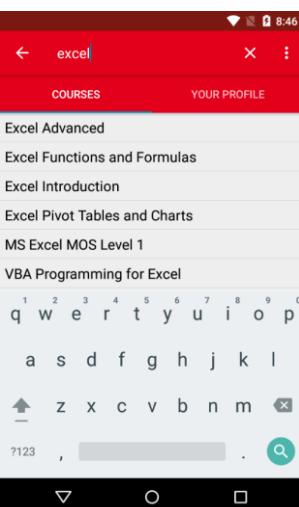
De typische Windows Phone-kenmerken zijn vooral zichtbaar in Figuur 52. Alle visuele elementen zijn vlak, maar geven wel aan welke mogelijkheden eronder verdoken zitten.

De *view lifecycle* werkt vrij eenvoudig op Windows Phone. Wanneer een scherm ingeladen wordt, wordt het *Loaded* event opgeroepen. De ontwikkelaar heeft de keuze om methodes toe te voegen voor de events *On-NavigatedTo* en *OnNavigatedFrom* die nadien opgeroepen worden.

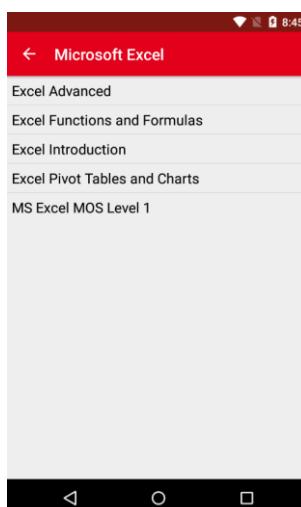
### 9.13.3 Android



Figuur 55: voorbeeld van Material Design in de Android app



Figuur 53: voorbeeld van Material Design in de Android app



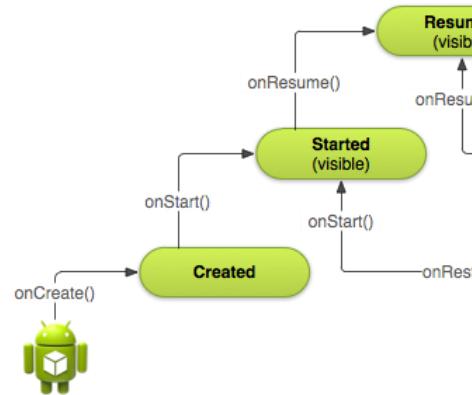
Figuur 54: voorbeeld van Material Design in de Android app

Google kwam enkele jaren geleden met een kleine revolutie op de proppen toen ze de grootste wijziging aan Android sinds Android voorstellen: Material Design. Dit is een uitgebreid uitgewerkte designfilosofie die op elk type software-interface toegepast kan worden. Het kenmerkt zich door visuele elementen in te delen in lagen die op elkaar liggen. Om dit te realiseren wordt er vaak

gebruik gemaakt van 3D-effecten zoals schaduwen en animaties. Enkele toepassingen hiervan zijn Figuur 53, Figuur 54 en Figuur 55.

De schermen in een Android app heten *activities*. Een activity kan op zich al heel wat elementen bevatten, maar kan ook opgebouwd zijn uit een of meerdere *fragments* die dan later opnieuw gebruikt kunnen worden in andere activities.

De lifecycle van een activity is beschreven in Figuur 56. Die van een fragment is weergegeven in Figuur 57.

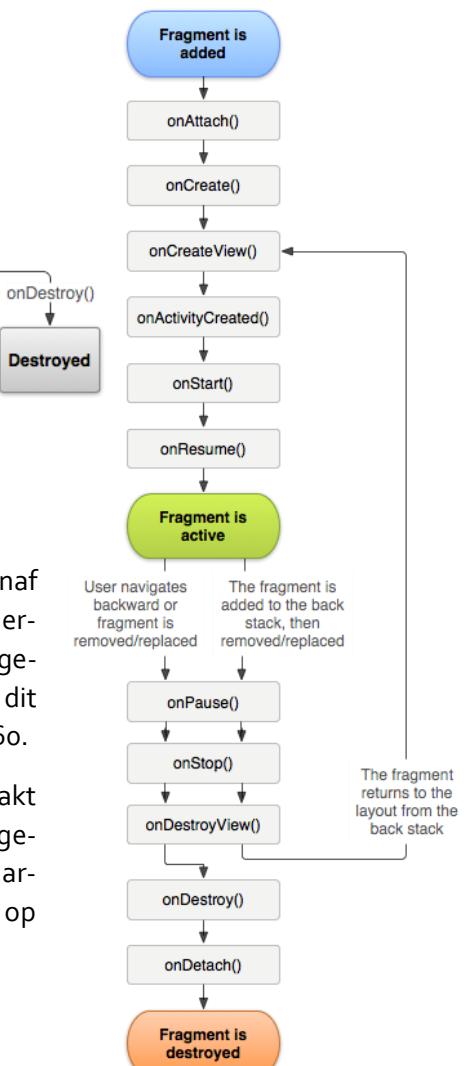


Figuur 56: lifecycle van een activity (Google Inc.)

#### 9.13.4 iOS

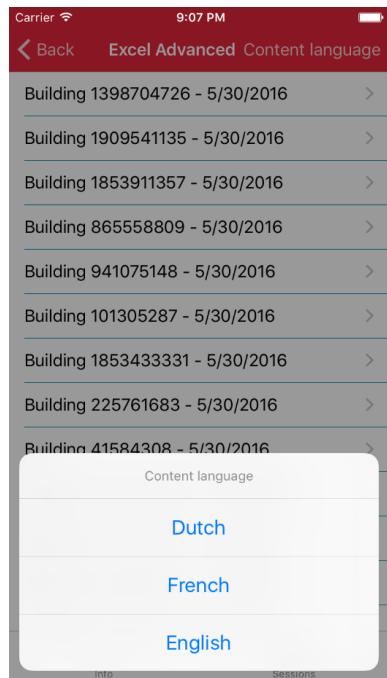
De 7<sup>de</sup> versie van iOS luidde het einde van skeuomorfisme<sup>10</sup> in. Vanaf deze versie koos Apple eveneens voor een vlak design waarbij onderscheid tussen elementen vooral aan de hand van kleuren teweeg gebracht werd. Ook in de applicatie voor RealDolmen Education werd dit toegepast. Voorbeelden hiervan zijn Figuur 58, Figuur 59 en Figuur 60.

De view lifecycle bij iOS is een geval apart. Elk scherm dat aangemaakt wordt, wordt enkel uit het geheugen verwijderd indien er geen vrije geheugenruimte meer beschikbaar is. De meeste schermen worden daardoor veel vaker opnieuw weergegeven dan bij hun tegenhangers op Android en op Windows Phone.

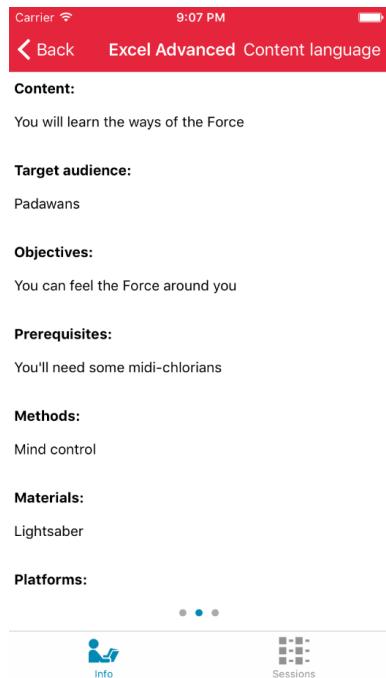


Figuur 57: lifecycle van een fragment (Google Inc.)

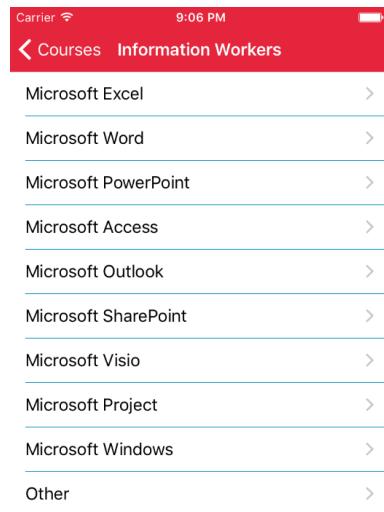
<sup>10</sup> Skeuomorfisme is een designfilosofie waarbij digitaal design bestaat uit nabootsing van elementen uit de analoge wereld. Een voorbeeld hiervan is een animatie van een pagina uit een digitaal document die omgeslagen wordt. Dit biedt geen functionele meerwaarde aan de toepassing, kost extra tijd bij de ontwikkeling en zou vervangen kunnen worden door kortere en duidelijker animaties.



Figuur 59: voorbeeld van vlak design op iOS



Figuur 58: voorbeeld van vlak design op iOS



Figuur 60: voorbeeld van vlak design op iOS

## 9.14 Ontwikkelomgeving

Naast het programmeren en ontwerpen zelf, kwam er ook wat werk kijken bij het installeren van de nodige software en hardware om met Xamarin aan de slag te kunnen gaan.

### 9.14.1 Microsoft Visual Studio en Xamarin voor Windows

Om een Windows Phone app te ontwikkelen is Microsoft Visual Studio 2013 of nieuwere op Windows 8.1 of nieuwere vereist. Dit is eveneens nodig om Xamarin apps te ontwikkelen op Windows.

De versie die RealDolmen aanbiedt aan stagairs is de gratis Visual Studio Community-editie. Deze gratis versie mist enkele mogelijkheden die wel in betalende versies aanwezig zijn: CPU- en geheugenverbruik meten tijdens het testen, code coverage van tests meten en aanwijzingen om snel door code te navigeren (CodeLens). Omdat studenten gratis toegang hebben tot alle betalende versies van Visual Studio, werd er toch geopteerd voor de Enterprise-versie van Visual Studio.

Een volgende vereiste om Xamarin apps te ontwikkelen zijn de Xamarin.Android en Xamarin.iOS SDK<sup>11</sup>'s. Deze voegen gedeeltelijke ondersteuning toe aan Visual Studio voor de ontwikkeling van Android en iOS apps met Xamarin. Tabel 13 geeft weer wat ondersteund is en wat niet. In Visual Studio ontbreekt IntelliSense<sup>12</sup> bij Android lay-out-bestanden. Dit zorgt voor het automatisch suggereren en aanvullen van code tijdens het typen. Bij de installatie van de SDK's wordt ook Xamarin Studio automatisch geïnstalleerd. Dit is een alternatieve ontwikkelomgeving waarin deze mogelijkheid wel beschikbaar is.

Type app/project	Ontwikkelplatform	Ondersteund in Visual Studio?	Ondersteund in Xamarin Studio?
Android	Windows	Ja	Ja
Android	OS X	Niet van toepassing	Ja
iOS	Windows	Gedeeltelijk	Nee
iOS	OS X	Niet van toepassing	Ja
Windows Phone	Windows	Ja	Nee
Windows Phone	OS X	Niet van toepassing	Nee
Gedeelde code (PCL)	Windows	Ja	Ja
Gedeelde code (PCL)	OS X	Niet van toepassing	Ja
Unit tests	Windows	Ja	Nee
Unit tests	OS X	Niet van toepassing	Nee

Tabel 13: ondersteunde ontwikkelconfiguraties (groen: beste ontwikkelervaring / meeste ondersteuning)

### 9.14.2 Xamarin Studio voor Mac

Om iOS apps te kunnen ontwikkelen met Visual Studio op Windows, is een verbinding naar een Mac met OS X nodig. De verbinding gebeurt via het SSH-protocol en op de Mac moeten Apple XCode en de Xamarin.iOS SDK geïnstalleerd zijn. Theoretisch gezien zou dit zonder problemen moeten werken, maar dit is in de praktijk niet altijd het geval. Zo was de MacBook waarover RealDolmen beschikt al wat ouder en niet echt performant en was het onmogelijk om verbinding over het beveiligde draadloze netwerk te maken. Er werd dan een rechtstreekse bekabelde verbinding gemaakt, maar er doken nog steeds problemen op.

Om zo weinig mogelijk tijd te verliezen werd de iOS app verder grotendeels ontwikkeld in Xamarin Studio voor OS X. Het enige nadeel hieraan was dat ondersteuning voor de laatste versie van de programmeertaal,

<sup>11</sup> Software Development Kit: software benodigd om andere software te ontwikkelen. Zo is bijvoorbeeld de Xamarin.Android SDK nodig om Android-apps te ontwikkelen met Xamarin.

<sup>12</sup> Een latere update voegde veel betere ondersteuning voor IntelliSense toe, echter biedt Xamarin Studio nog steeds meer aanvullingen en suggesties aan.

C# 6, ontbrak. Het programma compileerde zonder fouten, maar de automatische suggesties en aanvullingen in Xamarin Studio werkten vaak niet.

#### 9.14.3 Broncodebeheer (VCS): TFS en Git

Om op zowel Windows als Mac te kunnen beschikken over de laatste versie van de broncode, is een version control system (VCS) aangewezen. Een VCS slaat broncode op als revisies. Telkens wanneer een ontwikkelaar een aanpassing maakt aan de code, wordt een nieuwe revisie opgeslagen. Het ene versiebeheersysteem biedt al wat meer mogelijkheden dan het andere. Zo is Git het meest veelzijdige en meest populaire systeem.

Mogelijkheden waarmee de meeste systemen zich onderscheiden zijn vaak de volgende:

- **Branching:** in plaats van revisies achtereenvolgens op te slaan, wordt er gewerkt met vertakkingen. Zo kan de ontwikkelaar verder werken aan zijn versie van de code terwijl een andere ontwikkelaar telkens nieuwe revisies opslaat bovenop zijn eigen aanpassingen. Later kunnen beide versies worden samengevoegd.
- **Efficiëntie bij opslag:** bij nieuwe revisies zijn vaak niet alle bestanden aangepast. Het heeft dan ook geen zin om een nieuwe kopie van ongewijzigde bestanden op te slaan. Enkel de wijzigingen sinds de vorige revisie moeten ergens bijgehouden worden. Het ene systeem gaat hier zuiniger mee om dan het andere.

Bij RealDolmen worden de meeste projecten opgezet met Microsoft Team Foundation Service als VCS. Dit bevat beperkte ondersteuning voor branching en wordt uitsluitend ondersteund op Windows. Toen voor de iOS app werd overgestapt naar Xamarin Studio voor Mac, was het dan ook nodig om dit te vervangen door een andere VCS. Omdat integratie met Visual Studio Team Services vereist was, was de enige andere optie Git.

#### 9.14.4 Continue integratie met Visual Studio Team Services (VSTS)

VSTS is een gratis dienst van Microsoft die hulpmiddelen voor de scrum-methode aanbiedt om teams van softwareontwikkelaars efficiënter te laten werken. Naast de klassieke hulpmiddelen zoals een werk bord om taken te verdelen, biedt het ook continue integratie aan.

Continue integratie (CI) is het principe waarbij elke coderevisie uit een VCS gecompileerd en getest wordt. Ontwikkelaars stellen eerst tests op (zoals bij 9.12 TESTING) en configureren nadien een CI-pakket om deze tests automatisch uit te voeren. Het resultaat is een efficiëntere werking waarin bugs sneller opgelost worden. De volledige cyclus wordt overlopen in Figuur 61. Deze manier van werken wordt vaak de *DevOps*-manier genoemd.

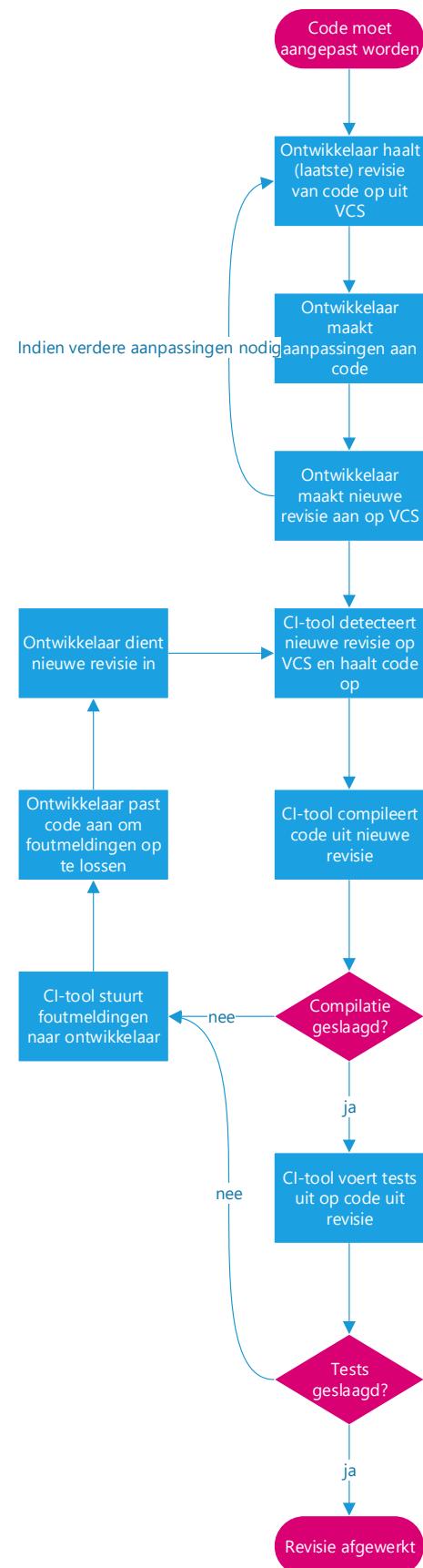
Zoals vermeld bij 9.14.2 XAMARIN STUDIO VOOR MAC kunnen iOS apps enkel op een Mac gecompileerd worden. De Windows Phone app kan daarentegen enkel op Windows met Visual Studio gecompileerd worden. Ook bij continue integratie is dit het geval. Om dit te realiseren, moeten minstens twee *build agents* ingesteld worden die de compilaties voor zich nemen:

- **Build agent met Windows:** compileert de Windows Phone app en de unit tests, voert nadien alle unit tests uit
- **Build agent met OS X:** compileert de Android en iOS apps

Voor de build agent met Windows kan een agent gebruikt worden die automatisch tijdelijk verleend wordt door VSTS. Microsoft stelt maandelijks 240 minuten beschikbaar om gratis gebruik te maken van dergelijke build agents.

Voor de OS X build agent moet het ontwikkelteam zelf een toestel beschikbaar stellen en configureren. Hiervoor werd de MacBook van RealDolmen ingesteld. Bij elke build maakt VSTS automatisch verbinding en wordt de build hierop uitgevoerd. De ontwikkelaars kunnen ondertussen via VSTS meteen opvolgen hoe het compileren verloopt.

Het is ook mogelijk om de Android app door de Windows build agent te laten afhandelen, maar door dit op de OS X build agent in te stellen, worden er gratis buildminuten gespaard.



Figuur 61: ontwikkelingscyclus bij gebruik van VCS en CI

## 9.15 Afhankelijkheden

De applicatie van RealDolmen Education maakt gebruik van bestaande onderdelen die publiekelijk beschikbaar zijn. Alle gebruikte afhankelijkheden zijn volledig gratis en zonder verplichtingen te gebruiken voor elk type project. Er werden twee soorten afhankelijkheden geïntroduceerd.

NuGet packages zijn codebibliotheeken die geïnstalleerd kunnen worden via de NuGet package manager ingebouwd in Microsoft Visual Studio. NuGet packages kunnen meestal in heel wat soorten projecten geïntegreerd worden (web apps, mobiele apps, desktop apps...)

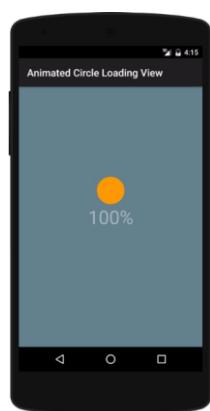
Xamarin Components zijn componenten die zowel code als design-bestanden kunnen bevatten. Deze componenten zijn uitsluitend beschikbaar voor apps ontwikkeld met het Xamarin-platform. Components voegen functionaliteiten toe aan een Xamarin app en kunnen uitsluitend voor Xamarin.Android, Xamarin.iOS, Xamarin.Forms, Windows Phone of combinaties hiervan beschikbaar zijn.

Sommige afhankelijkheden zijn zelf ook afhankelijk van enkele andere NuGet packages. Deze worden hier niet vermeld aangezien ze niet rechtstreeks in de code van het stageproject gebruikt worden.

Bij elke afhankelijkheid worden de doelprojecten vermeld waarin ze gebruikt worden. Soms zijn dit meer projecten dan de eigenlijke projecten die de afhankelijkheid gebruiken. De oorzaak hiervan is dat referenties naar de afhankelijkheden in de gedeelde code soms moeten gepropageerd worden naar de Android en de iOS app. Dit zorgt ervoor dat code uit de gedeelde projecten automatisch vervangen wordt door platformspecifieke implementaties.

### 9.15.1 Xamarin Components

#### 9.15.1.1 *AnimatedCircleLoadingView*



<b>Bron:</b>	<a href="https://components.xamarin.com/view/AnimatedCircleLoading-View/">https://components.xamarin.com/view/AnimatedCircleLoading-View/</a>
<b>Doelprojecten:</b>	Android app

Voor Windows Phone en iOS bestaan ingebouwde componenten die een indicatie van laden kunnen weergeven. Zo weet de gebruiker dat hij of zij even geduld moet uitoefenen terwijl de nodige gegevens ingeladen of verzonden worden. In de Android app werd deze Xamarin Component gebruikt omdat de ingebouwde component minder duidelijk was. Deze component kan zowel een loading indicator laten zien met een percentage als een zonder percentage.

Figuur 62: voorbeeld van de *AnimatedCircleLoadingView* component

#### 9.15.1.2 *Xamarin.Auth*

<b>Bron:</b>	<a href="https://components.xamarin.com/view/xamarin.auth/">https://components.xamarin.com/view/xamarin.auth/</a>
<b>Doelprojecten:</b>	Android app, iOS app

Bij het afhandelen van de authenticatie treden verschillende platformspecifieke fasen op. Zo moet de app een scherm laten zien met daarin de inlogpagina van de Security Token Service. Daarna moet de app ook het token kunnen afzonderen uit een URL. Deze code kan niet gedeeld worden over de platformen heen en

Xamarin.Auth bevat implementaties hiervan voor Android en voor iOS. In 9.9 AUTHENTICATIE wordt dieper ingegaan op de OAuth flow waarbij deze component gebruikt werd.

### 9.15.2 NuGet packages

#### 9.15.2.1 Akavache

Bron:	<a href="https://github.com/akavache/Akavache">https://github.com/akavache/Akavache</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Akavache is een package geschreven door Paul Betts voor Xamarin en Windows Phone/Universal apps om gegevens op een toestel te cachen. Het maakt gebruik van een SQLite-gegevensbank om gegevens in JSON-formaat op te slaan en op te halen. Voor het omzetten van en naar JSON wordt Json.NET (zie 9.15.2.11) gebruikt. Akavache regelt automatisch de encryptie van gevoelige gegevens, het beheer van de database en het opruimen van verouderde gegevens. Meer informatie over hoe dit package gebruikt werd, is terug te vinden bij 9.8 CACHING.

#### 9.15.2.2 Autofac

Bron:	<a href="http://autofac.org/">http://autofac.org/</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Autofac is een IoC<sup>13</sup>-container compatibel met elk type project op het .NET-platform. Dit zorgt ervoor dat een ontwikkelaar klassen in code op een losse manier aan elkaar kan koppelen en verhoogt de aanpasbaarheid van de code. Meer informatie over dit ontwerppatroon is terug te vinden bij 9.7.2 DEPENDENCY INVERSION.

#### 9.15.2.3 Base Class Libraries Build Components

Bron:	<a href="https://www.nuget.org/packages/Microsoft.Bcl.Build/">https://www.nuget.org/packages/Microsoft.Bcl.Build/</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Deze componenten voegen functies toe aan het compilatieproces waardoor projecten met portable class libraries (zie 9.6.2 GEDEELDE CODE) succesvol gecompileerd kunnen worden. Deze NuGet package dient enkel toegevoegd te worden indien er waarschuwingen verschijnen in de logboeken van het compilatieproces.

#### 9.15.2.4 Base Class Libraries Portability Pack

Bron:	<a href="https://www.nuget.org/packages/Microsoft.Bcl">https://www.nuget.org/packages/Microsoft.Bcl</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Deze codebibliotheek van Microsoft zorgt ervoor dat code uit nieuwere versies van het .NET-platform ook gebruikt kunnen worden in projecten die met een oudere versie van het .NET framework werken. Dit package is nodig bij Windows Phone 8.1 apps en Xamarin apps die gebruik maken van klassen en interfaces zoals Tuple, IProgress, Task...

<sup>13</sup> Inversion of Control

### 9.15.2.5 Common Service Locator

Bron:	<a href="https://commonservicelocator.codeplex.com/">https://commonservicelocator.codeplex.com/</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Deze codebibliotheek van Microsoft is een implementatie van het service location ontwerppatroon dat eveneens door Microsoft uitgetekend werd (Microsoft Corp., 2007). Dit patroon wordt als hulpmiddel gebruikt bij dependency injection (9.7.2 DEPENDENCY INVERSION). Het laat ontwikkelaars toe om dependency injection toe te passen op plaatsen in code waar constructor injection of property injection onmogelijk is.

### 9.15.2.6 Connectivity Plugin for Xamarin and Windows

Bron:	<a href="https://github.com/jamesmontemagno/Xamarin.Plugins/tree/master/Connectivity">https://github.com/jamesmontemagno/Xamarin.Plugins/tree/master/Connectivity</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

De code die nodig is om internetconnectiviteit te controleren is op elk mobiel platform anders. Deze plug-in voor Xamarin en Windows apps handelt de platformspecifieke code af en zorgt ervoor dat ook in de gedeelde code van deze functies gebruik gemaakt kan worden. Bij het compileren wordt de onderliggende code automatisch vervangen door een platformspecifieke variant. De plug-in wordt onderhouden door Xamarin zelf. Bij 9.8 CACHING wordt verder verduidelijkt waarom deze functie nodig is.

### 9.15.2.7 Fusillade

Bron:	<a href="https://github.com/paulcbetts/Fusillade">https://github.com/paulcbetts/Fusillade</a>
Doelprojecten:	Gedeelde code

Paul Betts' Fusillade is een krachtig hulpmiddel om een mobiele applicatie efficiënter te laten omgaan met bandbreedte. Mobiele applicaties moeten heel wat gegevens opvragen van het web, maar hebben niet altijd een stabiele internetverbinding tot hun beschikking.

Om de wachttijd van een gebruiker te beperken, kunnen sommige gegevens speculatief opgevraagd worden. Dit wil zeggen dat de applicatie voorspelt welke gegevens de gebruiker zal raadplegen en deze reeds in de achtergrond gaat ophalen.

Een gevolg hiervan kan zijn dat de gegevensoverdrachten op de achtergrond de gegevensoverdracht op de voorgrond belemmeren. Als de applicatie bezig is met heel wat gegevens op te halen, maar de gebruiker plots andere gegevens opvraagt, dan moet de applicatie voorrang geven aan dit laatste.

Een ander gevolg kan zijn dat een applicatie dezelfde gegevens meerdere keren gaan opvragen.

Fusillade zorgt ervoor dat ontwikkelaars gebruik kunnen maken van dit principe, zonder de nadelen. Het zorgt automatisch voor deduplicatie, prioritering en speculatieve overdrachten. De ontwikkelaar heeft hier zelf nog steeds controle over, maar Fusillade regelt zelf intern hoe dit allemaal verloopt.

### 9.15.2.8 HTTP Client Libraries

Bron:	<a href="https://www.nuget.org/packages/Microsoft.Net.Http">https://www.nuget.org/packages/Microsoft.Net.Http</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Dit package van Microsoft zorgt ervoor dat ontwikkelaars in elk .NET-project gebruik kunnen maken van dezelfde API's voor de *HttpClient*. Deze laatste is de meest voorkomende manier om te communiceren met

een API op het web. Zonder deze client libraries zijn niet alle functies van de *HttpClient* beschikbaar in projecten met Xamarin of Windows Phone. Wanneer deze libraries al dan niet gebruikt moeten worden is toch niet altijd duidelijk. Dit werd dan ook verduidelijkt in een blogpost op de Windows Blog (Nabar, 2015).

#### 9.15.2.9 IdentityModel

Bron:	<a href="https://github.com/IdentityModel/IdentityModel">https://github.com/IdentityModel/IdentityModel</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Deze package is ook wel bekend als ThinkTecture IdentityModel, maar de ontwikkelaars van dit project werken niet langer voor het bedrijf ThinkTecture waardoor de naam onlangs vereenvoudigd werd. Het bevat functies die helpen met de afhandeling van de OAuth flow waar dieper op ingegaan wordt in 9.9 AUTHENTIFICATIE.

#### 9.15.2.10 JimBobBennett.MvvmLight.AppCompat

Bron:	<a href="https://github.com/jimbobbennett/JimBobBennett.MvvmLight.AppCompat">https://github.com/jimbobbennett/JimBobBennett.MvvmLight.AppCompat</a>
Doelprojecten:	Android app

Veel van de functies die de MVVM Light Toolkit (zie ook 9.15.2.15 MVVM LIGHT TOOLKIT en 9.6.1 MVVM) aanbiedt, zijn niet compatibel met de Android Support Libraries. Jim Bob Bennett en Samuel Debruyn hebben daarom een NuGet package ontwikkeld dat de brug maakt tussen beide onderdelen. Het breidt de mogelijkheden van Laurent Bugnion's MVVM Light Toolkit uit naar projecten die gebruik maken van deze Support Libraries (zie ook 9.15.2.20 XAMARIN ANDROID SUPPORT LIBRARY: APPCOMPAT (v7)).

#### 9.15.2.11 Json.NET

Bron:	<a href="http://www.newtonsoft.com/json">http://www.newtonsoft.com/json</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

Newtonsoft JSON.NET is het populairste framework om gegevens in JSON<sup>14</sup>-formaat te verwerken in .NET. JSON is een gestandaardiseerd gegevensformaat en wordt vaak gezien als de opvolger van XML<sup>15</sup> om gegevens via het web te verzenden aangezien het veel minder tekens en dus ook bandbreedte vereist. JSON.NET maakt het omzetten van gegevens van en naar JSON doodeenvoudig en werkt in elk type project.

#### 9.15.2.12 Messaging Plugin for Xamarin and Windows

Bron:	<a href="https://github.com/cjlotz/Xamarin.Plugins/blob/master/Messaging/README.md">https://github.com/cjlotz/Xamarin.Plugins/blob/master/Messaging/README.md</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Ook deze plug-in dient om platformspecifieke code te vermijden. De applicatie voorziet de mogelijkheid om een lesgever een e-mail te sturen. Wanneer de gebruiker op de knop hiervoor drukt, wordt zijn/haar standaard e-mail app geopend en is het adres en het onderwerp van de e-mail reeds ingevuld. Dit zou heel wat code vereisen per mobiel platform, maar dankzij deze NuGet package kan deze functie in gedeelde code opgeroepen worden en is platformspecifieke code niet nodig.

<sup>14</sup> JavaScript Object Notation

<sup>15</sup> eXtensible Markup Language

### 9.15.2.13 ModernHttpClient

Bron:	<a href="https://github.com/paulcbetts/ModernHttpClient">https://github.com/paulcbetts/ModernHttpClient</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

De standaard implementatie van de *HttpClient* (zie ook 9.15.2.8) in Xamarin-projecten gebruikt onderliggend een component uit het Mono Project<sup>16</sup>. Deze component is veel minder performant dan de gelijkaardige componenten die standaard in Android en iOS zitten. ModernHttpClient van Paul Betts vervangt deze onderliggende component dan ook en vormt een brug naar *NSURLSession* bij iOS apps en *OkHttp* bij Android apps. Zo kunnen ontwikkelaars gebruik maken van de standaard *HttpClient* en toch beroep doen op de performantie van deze platformspecifieke implementaties.

### 9.15.2.14 Moq

Bron:	<a href="https://github.com/moq/moq4">https://github.com/moq/moq4</a>
Doelprojecten:	Unit tests

Moq is een mocking framework voor het .NET-platform. Het staat ontwikkelaars toe om abstracties te gebruiken tijdens het testen. Zonder een dergelijk framework zou de ontwikkelaar zelf een implementatie van deze abstractie moeten schrijven die specifiek gebruikt wordt tijdens het testen. Hier wordt dieper op in gegaan bij 9.12.1 MOCKING library: Moq.

### 9.15.2.15 MVVM Light Toolkit

Bron:	<a href="http://www.mvvmlight.net/">http://www.mvvmlight.net/</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app, unit tests

MVVM Light is een codebibliotheek van Laurent Bugnion, oorspronkelijk bedoeld om het MVVM<sup>17</sup>-ontwerp-patroon te faciliteren bij projecten die gebruik maken van Silverlight en/of WPF. Ondertussen is dit populaire package ook beschikbaar gemaakt voor heel wat andere soorten .NET-projecten zoals Xamarin en Windows Phone apps. Het gebruik van deze toolkit wordt verder verduidelijkt bij 9.6.1 MVVM.

### 9.15.2.16 PCLCrypto

Bron:	<a href="https://github.com/AArnott/PCLCrypto">https://github.com/AArnott/PCLCrypto</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Bij de authenticatie (zie ook 9.9 AUTHENTICATIE) wordt een willekeurig getal gebruikt om te voorkomen dat derden het authenticatiemechanisme kunnen onderscheppen en beïnvloeden. Dit willekeurig getal zou ge-genererd kunnen worden met de klasse *Random* uit het .NET Framework, maar kwaadwillige gebruikers zouden na kort onderzoek eenvoudig de gegenereerde willekeurige getallen kunnen voorspellen en zo als-nog de authenticatiegegevens onderscheppen. Daarom wordt er beroep gedaan op de cryptografische mogelijkheden van de processor van het toestel om het willekeurige getal te genereren. Microsoft voorziet hier voor Windows Phone zelf API's voor. Bij Xamarin apps kan beroep gedaan worden op dit NuGet package om ook daar de veiligheid van de authenticatie te verhogen.

<sup>16</sup> Open source-variant van Microsoft's .NET Framework

<sup>17</sup> Model-View-ViewModel

### 9.15.2.17 Refit

Bron:	<a href="http://paulbetts.github.io/refit/">http://paulbetts.github.io/refit/</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Paul Betts' Refit vereenvoudigt het schrijven van code om een RESTful API aan te spreken. Zo goed als elke mobiele app maakt wel gebruik van een webservice en deze service wordt vaak in het REST<sup>18</sup>-formaat aangeboden. Heel vaak moeten ontwikkelaars repetitieve code schrijven om diensten van dergelijke API's te gebruiken. Refit zorgt ervoor dat ze enkel nog een omschrijving van deze API moeten ingeven als een interface en handelt zelf alle repetitieve code af. Het voegt wat meer abstractie aan de code toe, wat de mogelijkheid tot testen bevordert.

### 9.15.2.18 Share Plugin for Xamarin and Windows

Bron:	<a href="https://github.com/jguerl/SharePlugin">https://github.com/jguerl/SharePlugin</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Op elk mobiel platform is de code om een browser te openen anders. Zo wordt bij Android gebruik gemaakt van een Intent terwijl Windows Phone en iOS rechtstreeks een URL kunnen openen. Om meer code te delen en geen platformspecifieke code hiervoor te moeten voorzien, werd dit NuGet package geschreven. Zo kunnen functies zoals het oproepen van de browser, het delen van een link en tekst kopiëren naar een klembord in de gedeelde code geplaatst worden.

### 9.15.2.19 Xamarin.Insights

Bron:	<a href="https://www.xamarin.com/insights">https://www.xamarin.com/insights</a>
Doelprojecten:	Gedeelde code, Android app, iOS app, Windows Phone app

Xamarin.Insights diende om crashes, foutmeldingen en andere logboeken bij een mobiele applicatie te verzamelen. De meldingen werden automatisch naar een centrale dienst doorgestuurd. Ontwikkelaars konden zich dan aanmelden en in een handig overzicht nakijken op welke types toestellen er zich problemen met hun applicaties voordeden en waar de oorzaak hiervan lag. In 2015 heeft Microsoft de dienst HockeyApp overgenomen en in 2016 volgde Xamarin. HockeyApp biedt quasi dezelfde functies als Xamarin.Insights en na beide overnames werden de diensten samengevoegd. Er wordt dieper in gegaan op deze functie bij 9.10 LOGGING.

### 9.15.2.20 Xamarin Android Support Library: AppCompat (v7)

Bron:	<a href="https://www.nuget.org/packages/Xamarin.Android.Support.v7.AppCompat/">https://www.nuget.org/packages/Xamarin.Android.Support.v7.AppCompat/</a>
Doelprojecten:	Android app

Veel Android-toestellen werken met verouderde versies van het Android besturingssysteem. In principe zou dit app ontwikkelaars verhinderen van de functies te gebruiken die geïntroduceerd werden in latere versies van Android. Om aan dit probleem tegemoet te komen, heeft Google de Android Support Libraries ontwik-

---

<sup>18</sup> Representational State Transfer: dit is een vorm van architectuur van een dienst op het web die gegevens aanbiedt gebaseerd op onderzoek door Roy Fielding (Fielding, 2000). Het zorgt ervoor dat webdiensten een uniforme en voor-spelbare structuur krijgen waardoor softwareontwikkelaars zich niet moeten verdiepen in de documentatie van een specifieke webdienst.

keld. Ontwikkelaars kunnen hierop beroep doen om hun applicatie compatibel te houden met oudere versies van Android. De Support Library zal automatisch ontbrekende functionaliteiten toevoegen op toestellen die nog niet met oudere versies van Android werken zodat noch de ontwikkelaar, noch de gebruiker, hier iets van merkt.

Dit package stelt de Android Support Library beschikbaar voor projecten op basis van Xamarin.Android.

#### 9.15.2.21 *Xamarin Android Support Library: Design*

Bron:	<a href="https://www.nuget.org/packages/Xamarin.Android.Support.Design/">https://www.nuget.org/packages/Xamarin.Android.Support.Design/</a>
Doelprojecten:	Android app

Het doel van dit package is gelijkaardig aan dat van 9.15.2.20 XAMARIN ANDROID SUPPORT LIBRARY: APPCOMPAT (V7). Bij Android 5.0 Lollipop introduceerde Google een nieuwe filosofie voor visueel ontwerp die *Material Design* heet. Het was de grootste verandering in Android sinds het ontstaan van Android zelf en Google raadde dan ook iedereen aan om van dan af aan apps te ontwikkelen die deze filosofie incorporeren en er daardoor beter, moderner en gebruiksvriendelijker uitzien en werken. Deze Support Library maakt het design ook beschikbaar voor toestellen die op een versie ouder dan Android 5.0 draaien. Er wordt verder ingegaan op dit design bij 9.13.3 ANDROID.

Het package maakt de Support Library met Material Design beschikbaar voor apps die van Xamarin.Android gebruik maken.

## 9.16 Broncode

Met dank aan Tom Knockaert van RealDolmen Education, werd de broncode beschikbaar gesteld via GitHub. Deze versie van de broncode bevat geen verwijzingen naar de API van RealDolmen Education en kan dus niet gebruikt worden om de applicatie te laten draaien.

De broncode is beschikbaar op <https://github.com/SamuelDebruyn/education-xamarin-app>

## 9.17 Presentaties

### 9.17.1 MS Community

[www.realdolmen.com](http://www.realdolmen.com)



**INHOUD**

- Opdracht: korte omschrijving
- Voorbereidende analyses
- Werkmethode
- Ervaringen

**OPDRACHT**

- RealDolmen Education
- Doelstelling
  - Website is niet responsive → niet bruikbaar op mobile
  - Aanwezigheid in app stores



**OPDRACHT**

- Mobile app met Xamarin
  - Android
  - iOS
  - Windows Phone
- Mobile app met UWP
  - Desktop
  - Tablet
  - Mobile
- Functies:
  - Opleidingen weergeven, zoeken
  - Inschrijven, feedback geven, contact opnemen
  - 3 talen: NL, FR, EN



**VOORBEREIDENDE ANALYSES**

- Technische analyse
  - Overzicht van architectuur
  - API-documentatie
  - Security (OAuth)
  - Overzicht dependencies (nuget...)
- Eventueel extra analyses op vraag van hogeschool
  - PID / plan van aanpak
  - Requirements (functioneel / niet-functioneel)

**WERKMETHODE**

- Verschillende sprints
  - Sprint opstellen via VSTS
  - Sprint (2 weken)
  - Sprint review
- Source code control: TFS / git

**ERVARINGEN BIJ STAGE**

- Redelijke vrijheid
  - 40h/week, vrij in te plannen
  - Werken vanuit RealDolmen-kantoor naar keuze
- Veel nieuwe mensen leren kennen
- Je kan altijd ergens terecht met problemen
  - Mondeling
  - Lync
  - E-mail
- Op korte tijd veel meer bijgeleerd dan tijdens opleiding



**THANK YOU**

For more information:  
visit our website [WWW.REALDOLMEN.COM](http://WWW.REALDOLMEN.COM)

Follow us on:

Selected presentations are available on: 



## 9.17.2 Slotpresentatie RealDolmen

www.realdolmen.com

**REALDOLMEN**

STAGE REALDOLMEN EDUCATION  
SAMUEL DEBRUYN

June 5, 2016 | SLIDE 1

REALDOLMEN

### INHOUD

- Wie ben ik?
- Voorstelling van het project
- Aanpak
- Demo
- Verloop van het project
- Technisch overzicht
- Reflectie

June 5, 2016 | SLIDE 2

### HET PROJECT

- De klant: RealDolmen Education
- Mobiele app om aanbod website weer te geven op smartphones
  - Android
  - iOS
  - Windows Phone



REALDOLMEN

June 5, 2016 | SLIDE 3

REALDOLMEN

### DOELSTELLINGEN

- Snel en eenvoudig zelfde functies als op website, mobiel aanbieden
- Aanwezigheid in app stores
  - Google Play
  - App Store
  - Windows Store



June 5, 2016 | SLIDE 4

### SCOPE VAN HET PROJECT

- App voor Android 4.4 of nieuwer
- App voor Apple iPhone met iOS 8 of nieuwer
- App voor Windows Phone 8.1 of nieuwer
- Categorieën van opleidingen
- Subcategorieën
- Details van opleidingen
- Zoeken naar opleidingen
- Inschrijven voor opleiding
- Feedback geven
- Contact opnemen met RealDolmen Education
- Gegevens offline beschikbaar
- Huisstijl van RealDolmen integreren
- App te gebruiken in Nederlands, Engels en Frans
- Analyses voorafgaand aan project

June 5, 2016 | SLIDE 5

REALDOLMEN

### BUITEN SCOPE VAN HET PROJECT

- Definitie RealDolmen huisstijl
- API die gegevens aanlevert
- Publicatie van app
- Aanleveren van ondersteunende tools (hardware, software nodig om project te ontwikkelen)

### AANPAK



June 5, 2016 | SLIDE 7

REALDOLMEN

June 5, 2016 | SLIDE 8

REALDOLMEN

**SEMESTER 1: BACHELORPROEFS**

- Voorbereidende analyses
  - Project Initiatie Document
  - Requirements analyse
  - Technische analyse
  - Use cases + use case diagram
- Voorafgaand onderzoek
  - Onderzoek cross platform development
  - Onderzoek hybrid apps / native apps
  - Onderzoek Xamarin
  - Prototype
  - Documentatie API

JUNE 5, 2016 | SLIDE 9

REALDOLMEN

JUNE 5, 2016 | SLIDE 10

REALDOLMEN

**SPRINT 1**

- Ik wil als geïnteresseerde een splash screen zien**
- Installeer benodigde tools en opbouw projecten
  - Tools installeren en verbinden met MacBook
  - Splash screen Android aanmaken
  - Splash screen Windows Phone aanmaken
  - Hoofdscherm Android aanmaken
  - Hoofdscherm Windows Phone aanmaken
  - Base classes voor http client aanmaken
  - Huistijl op Android aanmaken en integreren
  - Huistijl op Windows Phone aanmaken en integreren
  - CI tools opzetten
  - Splash screen iOS aanmaken
  - Hoofdscherm iOS aanmaken
  - Huistijl op iOS aanmaken en integreren
  - iOS Development intro

- |  | Done | Stageproject Education Samuel\Sprint 2 |
|--|------|--|
| ■ Installeer benodigde tools en opbouw projecten   | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Tools installeren en verbinden met MacBook       | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Splash screen Android aanmaken                   | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Splash screen Windows Phone aanmaken             | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Hoofdscherm Android aanmaken                     | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Hoofdscherm Windows Phone aanmaken               | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Base classes voor http client aanmaken           | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Huistijl op Android aanmaken en integreren       | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Huistijl op Windows Phone aanmaken en integreren | Done | Stageproject Education Samuel\Sprint 1 |
| ■ CI tools opzetten                                | Done | Stageproject Education Samuel\Sprint 1 |
| ■ Splash screen iOS aanmaken                       | Done | Stageproject Education Samuel\Sprint 2 |
| ■ Hoofdscherm iOS aanmaken                         | Done | Stageproject Education Samuel\Sprint 2 |
| ■ Huistijl op iOS aanmaken en integreren           | Done | Stageproject Education Samuel\Sprint 2 |
| ■ iOS Development intro                            | Done | Stageproject Education Samuel\Sprint 2 |

**DEMO**

JUNE 5, 2016 | SLIDE 11

REALDOLMEN

JUNE 5, 2016 | SLIDE 12

REALDOLMEN

**SPRINT 1**

- Ik wil als geïnteresseerde een lijst van alle categorieën kunnen ophalen**
- Code om categorieën uit API te halen
  - Lijst voor categorieën aanmaken op Android
  - Lijst voor categorieën aanmaken op Windows Phone
  - Categorieën weergeven in lijst op Android
  - Categorieën weergeven in lijst op Windows Phone
  - Caching voorzien om categorieën te bewaren
  - Lijst voor categorieën aanmaken op iOS
  - Categorieën weergeven in lijst op iOS
  - Ik wil als geïnteresseerde een lijst van alle subcategorieën in een categorie k... Done Stageproject Education Samuel\Sprint 2
  - API calls voor subcategorieën
  - Subcategorieën weergeven op Android
  - Subcategorieën weergeven op iOS
  - Subcategorieën weergeven op Windows Phone

- Code om categorieën uit API te halen
- Lijst voor categorieën aanmaken op Android
- Lijst voor categorieën aanmaken op Windows Phone
- Categorieën weergeven in lijst op Android
- Categorieën weergeven in lijst op Windows Phone
- Caching voorzien om categorieën te bewaren
- Lijst voor categorieën aanmaken op iOS
- Categorieën weergeven in lijst op iOS
- Ik wil als geïnteresseerde een lijst van alle subcategorieën in een categorie k... Done Stageproject Education Samuel\Sprint 1
- API calls voor subcategorieën
- Subcategorieën weergeven op Android
- Subcategorieën weergeven op iOS
- Subcategorieën weergeven op Windows Phone

JUNE 5, 2016 | SLIDE 13

REALDOLMEN

JUNE 5, 2016 | SLIDE 14

REALDOLMEN

**SPRINT 2**

- Ik wil als geïnteresseerde kunnen zoeken naar een opleiding**
- Zoekbaar- en functie op Android
  - Zoekbaar- en functie op Windows Phone
  - Zoekbaar- en functie op iOS
  - API-calls voor zoeken naar opleidingen
  - Resultaten weergeven op Android
  - Resultaten weergeven op iOS
  - Resultaten weergeven op Windows Phone
  - Onderzoek nodig voor ontwikkeling
  - Onderzoek naar x-platform IBn
  - Tests schrijven

- Zoekbaar- en functie op Android
- Zoekbaar- en functie op Windows Phone
- Zoekbaar- en functie op iOS
- API-calls voor zoeken naar opleidingen
- Resultaten weergeven op Android
- Resultaten weergeven op iOS
- Resultaten weergeven op Windows Phone
- Onderzoek nodig voor ontwikkeling
- Onderzoek naar x-platform IBn
- Tests schrijven

JUNE 5, 2016 | SLIDE 15

REALDOLMEN

JUNE 5, 2016 | SLIDE 16

REALDOLMEN

**SPRINT 3**

- Ik wil als geïnteresseerde kunnen zoeken naar een opleiding**
- API-methodes voor opleidingen van categorie
  - Scherm voor opleidingen in categorie op Windows Phone
  - Scherm voor opleidingen in categorie op iOS
  - Scherm voor opleidingen in categorie op Android
  - Viewmodel voor subcategorie
  - Ik wil als gebruiker mezelf kunnen uitloggen
  - Uitlogknop en bindings op Android
  - Uitlogknop en bindings op Windows Phone
  - Uitlogknop en bindings op iOS

- |   | Done | Stageproject Education Samuel\Sprint 3 |
|---|------|--|
| ■ API-methodes voor opleidingen van categorie           | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Scherm voor opleidingen in categorie op Windows Phone | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Scherm voor opleidingen in categorie op iOS           | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Scherm voor opleidingen in categorie op Android       | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Viewmodel voor subcategorie                           | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Ik wil als gebruiker mezelf kunnen uitloggen          | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Uitlogknop en bindings op Android                     | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Uitlogknop en bindings op Windows Phone               | Done | Stageproject Education Samuel\Sprint 3 |
| ■ Uitlogknop en bindings op iOS                         | Done | Stageproject Education Samuel\Sprint 3 |

## SPRINT 4

<span style="color: #0070C0;">■</span> Ik wil als geïnteresseerde gedetailleerde informatie over een opleiding kunnen bekijken	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> API-methodes voor details van een opleiding	Done	Stageproject Education Samuel\Sprint 3
<span style="color: #0070C0;">■</span> Scherm voor opleiding op Android	Done	Stageproject Education Samuel\Sprint 4
<span style="color: #0070C0;">■</span> Scherm voor opleiding Windows Phone	Done	Stageproject Education Samuel\Sprint 4
<span style="color: #0070C0;">■</span> ViewModel voor opleiding	Done	Stageproject Education Samuel\Sprint 4
<span style="color: #0070C0;">■</span> Unit tests voor ViewModel	Done	Stageproject Education Samuel\Sprint 4
<span style="color: #0070C0;">■</span> Scherm voor opleiding op iOS	Done	Stageproject Education Samuel\Sprint 5

## SPRINT 5

<span style="color: #0070C0;">■</span> Ik wil als geïnteresseerde alle lessen van een opleiding kunnen bekijken	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> API-methodes om sessies op te halen	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> ViewModel voor opleiding bijwerken	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Lijst van sessies op Android	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Lijst van sessies op iOS	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Lijst van sessies Windows Phone	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Unit tests voor opleiding vm bijwerken	Done	Stageproject Education Samuel\Sprint 5

JUNE 5, 2016 | SLIDE 17

**REALDOLMEN**

JUNE 5, 2016 | SLIDE 18

**REALDOLMEN**

## SPRINT 4-5-6

<span style="color: #0070C0;">■</span> Ik wil als gebruiker contact kunnen opnemen met RealDolmen Education	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> API-methodes voor contact	Done	Stageproject Education Samuel\Sprint 3
<span style="color: #0070C0;">■</span> ViewModel voor contactpagina	Done	Stageproject Education Samuel\Sprint 3
<span style="color: #0070C0;">■</span> Contactformulier op Windows Phone	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Contactformulier op iOS	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Contactformulier op Android	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Ik wil als gebruiker feedback over een les kunnen geven	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> API-methodes voor feedback	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> ViewModel voor feedback	Done	Stageproject Education Samuel\Sprint 3
<span style="color: #0070C0;">■</span> Feedbackscherm op Windows Phone	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Feedbackscherm op iOS	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> Feedbackscherm op Android	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> Ik wil als gebruiker mij kunnen inschrijven voor een opleiding	Committed	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> ViewModel voor inschrijving	Done	Stageproject Education Samuel\Sprint 3
<span style="color: #0070C0;">■</span> API-methodes voor inschrijvingen	Done	Stageproject Education Samuel\Sprint 4
<span style="color: #0070C0;">■</span> Inschrijfscherm Windows Phone	Done	Stageproject Education Samuel\Sprint 5
<span style="color: #0070C0;">■</span> Inschrijfscherm op Android	Done	Stageproject Education Samuel\Sprint 6
<span style="color: #0070C0;">■</span> Inschrijfscherm op iOS	To Do	Stageproject Education Samuel\Sprint 6

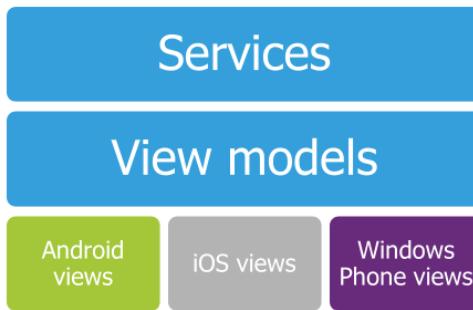
JUNE 5, 2016 | SLIDE 19

**REALDOLMEN**

JUNE 5, 2016 | SLIDE 20

**REALDOLMEN**

## ARCHITECTUUR



JUNE 5, 2016 | SLIDE 21

**REALDOLMEN**

JUNE 5, 2016 | SLIDE 22

**REALDOLMEN**

## REFLECTIE

- **Stagebedrijf**
  - Zeer waardevolle ervaring
  - Goede begeleiding
  - Vlotte communicatie
- **Zelfreflectie**
  - Op korte tijd enorm veel kennis opgedaan
  - Voorbereiding heeft nut bewezen
  - Nieuwe concepten:
    - iOS development & design
    - Model-View-ViewModel
    - Xamarin
  - **Verder uitgediept:**
    - Windows Phone development
    - Android development & design
    - Scrum
    - Continue integratie
  - **Doel behaald**

JUNE 5, 2016 | SLIDE 23

**REALDOLMEN**

JUNE 5, 2016 | SLIDE 24

**REALDOLMEN**

# BEDANKT!

REALDOLMEN

For more information:  
visit our website [WWW.REALDOLMEN.COM](http://WWW.REALDOLMEN.COM)

Follow us on:



Selected presentations are available on:



## 9.17.3 Slotpresentatie Odisee

O'dee

Xamarin app  
RealDolmen Education

Samuel Debruyn

3IT - optie Software Engineering

BRONSTED  
KU LEUVEN

### De klant

**RealDolmen: ICT integrator**

**RealDolmen Education: opleidingen voor zowel eigen als externe medewerkers**

We make ICT work for your business

Project-omschrijving  
Aanpak  
Xamarin  
Verloop van het project  
Technisch  
Demo  
Besluit & reflectie

Xamarin appRealDolmen Education 09/06/2016 3

### Projectomschrijving

- ➊ Alternatief voor website op smartphones
- ➋ Zichtbaarheid in applicatiemarktjes
- ➌ Eén mobiele app, meerdere platformen

Project-omschrijving  
Aanpak  
Xamarin  
Verloop van het project  
Technisch  
Demo  
Besluit & reflectie

Xamarin appRealDolmen Education 09/06/2016 5

### Aanpak

Project-omschrijving  
Aanpak  
Xamarin  
Verloop van het project  
Technisch  
Demo  
Besluit & reflectie

Xamarin appRealDolmen Education 09/06/2016 7

### Inhoud

- Klant- en projectomschrijving
- Aanpak en voorbereiding
- Xamarin
- Verloop van het project
- Technische concepten
- Demo
- Besluit en reflectie

Xamarin appRealDolmen Education 09/06/2016 2

### Projectomschrijving

Welcome bij RealDolmen Education.

Home | Contact | Brussel | Gedacht | Over ons

Zoek

Project-omschrijving  
Aanpak  
Xamarin  
Verloop van het project  
Technisch  
Demo  
Besluit & reflectie

Welkom bij RealDolmen Education.

Een bekendt dat u een opleiding moet volgen of op basis van een projectwerkje te bekijken.

Die pagina's zijn u overzichtelijk, gedetailleerd, concreet, georganiseerd en toegankelijk.

Tot slot is hiermee een opleiding die de recentste ontwikkelingen of praktische toepassingen in een cursus.

Nieuwe cursussen

- Agile for business analysis
- Agile for foundation
- Agile for foundation and practitioner
- APMS Learn IT

Xamarin appRealDolmen Education 09/06/2016 4

### Scope

In scope	Out of scope
<ul style="list-style-type: none"> <li>➊ Navigatie via (sub)categorieën en zoeken</li> <li>➋ Details van opleidingen met sessies &amp; data</li> <li>➌ Inschrijven, feedback geven, contact opnemen</li> <li>➍ Gegevens offline beschikbaar</li> <li>➎ Huisstijl van RealDolmen</li> <li>➏ Nederland, Engels en Frans</li> <li>➐ Analyses voorafgaand aan project</li> </ul>	<ul style="list-style-type: none"> <li>➊ Definitie huisstijl</li> <li>➋ Web API die gegevens aanlevert</li> <li>➌ Publicatie van app</li> <li>➍ Aanleveren van ondersteunende tools (hardware, software nodig om project te ontwikkelen)</li> </ul>

Xamarin appRealDolmen Education 09/06/2016 6

### Semester 1: bachelorproef

Q Voorbereidende analyses	Q Voorafgaand onderzoek
<ul style="list-style-type: none"> <li>• Project Initiate Document</li> <li>• Requirements analyse</li> <li>• Technische analyse</li> <li>• Use cases + use case diagram</li> </ul>	<ul style="list-style-type: none"> <li>• Onderzoek cross platform development</li> <li>• Onderzoek hybride apps / native apps</li> <li>• Onderzoek Xamarin</li> <li>• Prototype</li> <li>• Documentatie API</li> </ul>

Xamarin appRealDolmen Education 09/06/2016 8

## Oplossing: Xamarin

Xamarin.Android →	Android app met .NET
Xamarin.iOS →	iOS app met .NET
Windows Phone	
⊕ Native code op elk platform	
⊕ Native design op elk platform	
⊕ Code sharing	
⊖ Licentiekosten	

Xamarin app RealDolmen Education 09/06/2016 9

## Projectmatig & verloop

- 6 sprints van elk 2 weken
- Sprint planning & sprint review
- Steeds realistischere inschattingen

Relevante evenementen

Xamarin app RealDolmen Education 09/06/2016 10

### Sprint 1

- Ik wil als geïnteresseerde een **splash screen** zien.
- Ik wil als geïnteresseerde een **lijst van alle categorieën** kunnen ophalen.
- Ik wil als geïnteresseerde me kunnen **inloggen** en zo kenbaar maken als gebruiker.

Xamarin app RealDolmen Education 09/06/2016 11

### Sprint 2

- Ik wil als geïnteresseerde kunnen **zoeken** naar een opleiding.
- Ik wil als geïnteresseerde een **lijst van alle subcategorieën** in een categorie kunnen ophalen.

Xamarin app RealDolmen Education 09/06/2016 12

### Sprint 3

- Ik wil als gebruiker **feedback** over een les kunnen geven.
- Ik wil als geïnteresseerde alle **opleidingen van een subcategorie** kunnen ophalen.
- Ik wil als gebruiker mij kunnen **inschrijven** voor een opleiding.
- Ik wil als gebruiker mezelf kunnen **uitloggen**.

Xamarin app RealDolmen Education 09/06/2016 13

### Sprint 4

- Ik wil als geïnteresseerde **gedetailleerde informatie** over een opleiding kunnen bekijken.
- Ik wil als gebruiker mij kunnen **inschrijven** voor een opleiding.

Xamarin app RealDolmen Education 09/06/2016 14

### Sprint 5

- Ik wil als geïnteresseerde **alle lessen** van een opleiding kunnen bekijken.

Xamarin app RealDolmen Education 09/06/2016 15

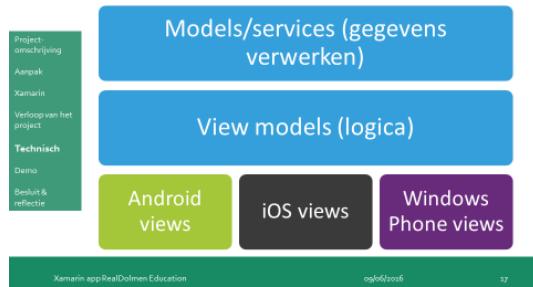
### Sprint 6

- Ik wil als gebruiker **contact kunnen opnemen** met RealDolmen Education.
- Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar **Nederlands**.
- Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar **Engels**.
- Ik wil als geïnteresseerde de taal van de applicatie kunnen aanpassen naar **Frans**.

Xamarin app RealDolmen Education 09/06/2016 16

## Architectuur

Gemiddeld 75% gedeelde code per app



## Technische concepten

**Ontwikkelomgeving:** continue integratie, Visual Studio & Xamarin Studio

**Testing:** ≈ 80% coverage met unit tests

**Design patterns:** MVVM, dependency inversion, observable-observer, builder, command...

**Caching:** gegevens offline beschikbaar via SQLite database

**Authenticatie:** volledig afgescheiden van app d.m.v. OAuth2 en Security Token Service

**Vertalingen:** gedeelde vertalingen voor 3 platformen

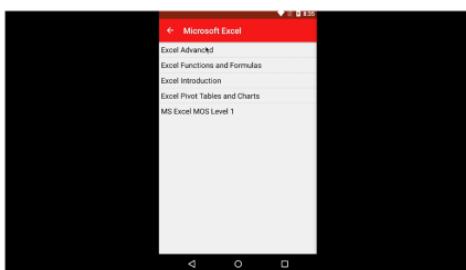
**Visueel ontwerp:** native design op elk platform (Material design, iOS 7(+), design, Metro/Modern UI)

## Demo: authenticatie



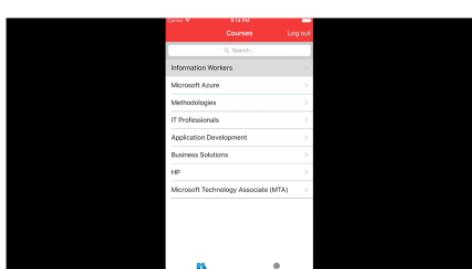
Xamarin app RealDolmen Education    09/06/2016    19

## Demo: contactfuncties



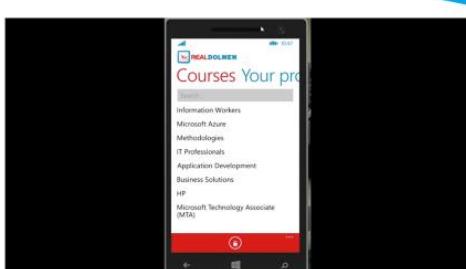
Xamarin app RealDolmen Education    09/06/2016    20

## Demo: zoekfunctie



Xamarin app RealDolmen Education    09/06/2016    21

## Demo: opleidingsdetails



Xamarin app RealDolmen Education    09/06/2016    22

## Besluit

Alle (niet-)functionele requirements geïmplementeerd

Solide architectuur, kwalitatief resultaat

Kan met minimale afwerking omgezet worden in gepubliceerd eindproduct

Xamarin is goede oplossing voor cross-platform apps

Project-  
omschrijving  
Aanpak  
Xamarin  
Verloop van het  
project  
Technisch  
Demo  
**Besluit &  
reflectie**

Xamarin app RealDolmen Education

09/06/2016

23

## Zelfreflectie

Voorbereiding heeft nut bewezen

Op korte tijd enorm veel kennis verworven

☒ Technisch

☒ Professioneel & projectmatig

Doel behaald

Aan de slag als Xamarin developer bij



Xamarin app RealDolmen Education

09/06/2016

24

## Vragen?

☒ <https://github.com/SamuelDebruyne/education-xamarin-app>

☒ <https://sa.muel.be/portfolio/realdolmen-education/>

Xamarin app RealDolmen Education

09/06/2016

25